ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

MANIPULATION OF VISUALLY RECOGNIZED OBJECTS USING DEEP LEARNING

Ph.D. THESIS

Ertuğrul BAYRAKTAR

Mechatronics Engineering Department

Mechatronics Engineering Doctorate Program

FEBRUARY 2018

ISTANBUL TECHNICAL UNIVERSITY ★ GRADUATE SCHOOL OF SCIENCE ENGINEERING AND TECHNOLOGY

MANIPULATION OF VISUALLY RECOGNIZED OBJECTS USING DEEP LEARNING

Ph.D. THESIS

Ertuğrul BAYRAKTAR
(518132007)

Mechatronics Engineering Department

Mechatronics Engineering Doctorate Program

Thesis Advisor: Assoc. Prof. Dr. Pınar BOYRAZ

FEBRUARY 2018

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

GÖRSEL TANINAN NESNELERİN DERİN ÖĞRENME KULLANILARAK HAREKET ETTİRİLMESİ

DOKTORA TEZİ

Ertuğrul BAYRAKTAR
(518132007)

Mekatronik Mühendisliği Anabilim Dalı

Mekatronik Mühendisliği Doktora Programı

Tez Danışmanı: Assoc. Prof. Dr. Pınar BOYRAZ

ŞUBAT 2018

Ertuğrul BAYRAKTAR, a Ph.D. student of ITU Graduate School of Science Engineering and Technology 518132007 successfully defended the thesis entitled "MANIPULATION OF VISUALLY RECOGNIZED OBJECTS USING DEEP LEARNING", which he/she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Pınar BOYRAZ** Istanbul Technical University	.....
Jury Members :	**Prof. Dr. Tülay YILDIRIM** Yıldız Technical University	.....
	Assoc. Prof. Dr. Hatice KÖSE Istanbul Technical University	.....
	Prof. Dr. Şeniz ERTUĞRUL Istanbul Technical University	.....
	Assist. Prof. Dr. Figen ÖZEN Haliç University	.....

Date of Submission : **1 February 2018**
Date of Defense : **28 February 2018**

To my family,

FOREWORD

This thesis was written for my Doctor of Philosophy degree in Mechatronics Engineering at Istanbul Technical University. I have integrated visual understanding into robotic manipulation semantically using deep neural networks within the scope of this thesis. To bridge the gap between real-world and simulation in robotic applications I have constructed a hybrid visual object dataset that is composed of real and synthetic images. I believe that the evidences will contribute the current literature.

I am grateful to following people and establishments for their responsibility, support and help. Firstly, I would like show my appreciation and respect to my supervisor Assoc. Prof. Dr. Pınar BOYRAZ for her encouragement, guidance and heartiness. Secondly, I would like to give my special thanks to my colleagues from Mechatronics Research and Education Center for sharing their knowledge, experience and valuable time. Additionally, I am very grateful to Cihat Bora YİĞİT for his co-operation and contribution during our complementary studies.

Finally, I would like to thank my wife Bilgenur BAYRAKTAR, my parents Recai & Hüsniye BAYRAKTAR and my sisters Aslı Büşra and Sena BAYRAKTAR for their constant supports, encouragements and goodwill during the time I studied.

February 2018

Ertuğrul BAYRAKTAR
(Mechanical Engineer)

TABLE OF CONTENTS

Page

ABBREVIATIONS

1D : 1-Dimensional
2D : 2-Dimensional
3D : 3-Dimensional
ADORESET : A Hybrid Dataset Towards Bridging The Gap Between Real And Simulation Environments for Robotics
AI : Artificial Intelligence
AMT : Amazon Mechanical Turk
ANN : Artificial Neural Network
ATE : Absolute Trajectory Error
AUC : Area Under The ROC Curve
BN : Batch Normalization
BRIEF : Binary Robust Independent Elementary Features
BRISK : Binary Robust Invariant Scalable Keypoints
CAD : Computer Aided Design
CNN : Convolutional Neural Network
COCO : Common Objects In Context
CONV : Convolution
CPU : Central processing Unit
DL : Deep Learning
DNN : Deep Neural Network
DOF : Degree of Freedom
DOG : Difference Of Gaussian
ELU : Exponential Linear Unit
F/T : Force/Torque
FAST : Features From Accelerated Segment Test
FC : Fully Connected
FLOPS : Floating Point Operation Per Second
FPN : Feature Pyramid Network
FREAK : Fast Retina Keypoint
GAN : Generative Adversarial Network
GB : Gigabyte
GHZ : Gigahertz
GLOH : Gradient Location Oriented Histogram
GSE : Gazebo Simulation Environment
GPS : Global Positioning System
GPU : Graphical Processing Unit
GUI : Graphical User Interface
HOG : Histogram Of Oriented Gradients
ILSVRC : ImageNet Large Scale Visual Recognition Challenge
IMU : Inertial Measurement Unit
IOU : Intersection Over Union
KF : Kalman Filter

LOG : Laplacian Of Gaussian
MAV : Micro Air Vehicle
MLP : Multi Layer Perceptron
MSER : Maximally Stable Regions
ORB : Oriented FAST and Rotated BRIEF
RELU : Rectified Linear Unit
RMS : Root Mean Square
RMSE : Root Mean Square Error
RNN : Recurrent Neural Network
ROC : Receiver Operating Characteristics
ROI : Region of Interest
ROS : Robot Operating System
RPM : Revolutions Per Minute
RPN : Region Proposal Network
SELU : Scaled Exponential Linear Unit
SIFT : Scale Invariant Feature Transform
SLAM : Simultaneous Localization and Mapping
SSD : Single-Shot Multi-Box Detector
SURF : Speeded-Up Robust Features
SVM : Support Vector Machine
SYNTHIA : Synthetic Collection Of Imagery and Annotations
TB : Terabyte
URDF : Universal Robot Description Files
VAE : Variational Autoencoder
VIA : Variable-impedance Actuator
VOC : Visual Object Classes
VRP : Variable Radius Pulley
VSA : Variable-stiffness Actuator
VSJ : Variable-stiffness Joint
YCB : Yale-CMU-Berkeley
YOLO : You Look Only Once

SYMBOLS

c	: Class confidence score
CE	: Regular cross-entropy loss
D	: Volume depth
E	: Expectation w.r.t. the empirical probability distribution
F	: Filter size
g	: Ground-truth parameters
G	: Derivative of images
H	: Volume height
H	: Hessian matrix
I	: Image
J	: Cost function
k	: Coefficient
K	: Filter depth (the number of filters)
l	: Predicted box
L	: Scale-space representation of the Laplacian function
m	: Weighted mean values of the pixels
M	: Magnitude of the gradient
N	: Number of matched default boxes
P	: Amount of zero-padding
PL	: Pooling layer output
r	: Minimum distance between the object and camera
R	: Response of the detector at each pixel
s	: Unit vector
S	: Amount of stride
S	: Sum of products
T	: Threshold value
t	: Time
w	: Angular velocity
W	: Volume width
x	: Pixel coordinates at the horizontal axis
x	: Input vector to the neural network
x_c, y_c, z_c	: Camera direction vectors
y	: Pixel coordinates at the vertical axis
$\bigtriangledown$	: Magnitude operation
α	: Sampling rotation angle
β	: Parameters to be learned
c_1	: Maximum distance between the object and camera
δ	: Partial derivative
ε	: Error value
γ	: Tunable focusing parameter
h_ω	: Predicted outputs
L_1	: L_1 norm

L_2 : L_2 norm
λ : Regularization coefficient
μ : Mean
σ : Standard deviation
ω : Weight vector

LIST OF TABLES

Page

LIST OF FIGURES

Page

MANIPULATION OF VISUALLY RECOGNIZED OBJECTS USING DEEP LEARNING

SUMMARY

People collect the greatest and most qualified data from their environment through vision systems. However, for a more complete and reliable perception, it is necessary to use corresponding data from other senses. Analogous to humans, robots collect data from the medium they are in via sensors. Object detection, recognition, and semantic value attribution are among the most recent research areas in robotics. The development of software and hardware technologies ensures the intelligence of everyday life. High-resolution, depth-sensing cameras, such as the Internet devices of objects, have made it possible to obtain multi-dimensional and large volume data. In parallel, robots have begun to be regarded as part of social life as well as the industrial field use. Human-robot interaction systems require high accuracy and speed in terms of real-time operation as content.

In the context of human-robot interaction, safe, fast, and capabilities with high performance/low error rates have become possible with the help of the advanced machine learning algorithms and the relevant hardware technologies for these algorithms. In intelligent manufacturing facilities, the robots are directly dependent on their hardware and software systems for their movements during their displacement, their ability to perform their assigned tasks at expected performance levels. Convolutional neural networks (CNNs) are trained for purposes such as object recognition, object boundary detection, object segmentation, semantic linkage.

CNNs are a generic name given to specialized artificial neural network models that contain a certain number of hidden layers, with some extraordinary architecture, parameter update methods, and activation functions. Deep CNNs trained using large amount of data that give results with minimal error values that are better than human performance regarding recognizing objects in the data set they are trained in, determining bounding-box coordinates surrounding the objects, and segmenting. Object localization and recognition operations for applications in the field of robotics are inadequate concerning semantic information extraction and object-based relationships. For this reason, the class to which the object belongs is assigned attributes beyond the class labels, so that the algorithms can infer from the semantic content.

In this thesis, the performances of the conventional visual feature detector and descriptor methods are analyzed in detail. In addition to the ordinary criteria such as speed, performance, and matching feature per image as performance criteria, we also took the distance between matching attributes, the number of correct matching attributes and the angular orientation difference between matching points into account during performance comparisons. In the experiments, a query dataset consisting of 127 template images was conducted matches with a dataset consisting of 3090

images. As a result of this study, it has been shown that some conventional feature extraction methods yield acceptable levels when accuracy is considered. However, no high-accuracy combination suitable for real-time operation has been achieved. From this aspect, manipulating visually recognized objects within this thesis with a robotic mechanism has been executed using deep learning methods. As a robotic mechanism, a structure consisting of a variable radius pulley-variable stiffness joint system is integrated into the experimental environment we call it the *Deep Table*.

In this thesis, an image data set named *ADORESet* was built to bridge the gap between the real world conditions and the simulation environments for use in robot vision studies. *ADORESet* consists of 30 categories consisting 2500 real, 750 synthetic images in each class, which are manually labeled and bounding-boxes are also specified by hand. We use *VGGNet* to perform the object recognition process and *RetinaNet* to determine the object locations when moving objects.

In this thesis, an alternative pooling layer is suggested to extend the literature. This method, called *Smart-pooling*, processes the relevant filter by taking the values of large or small pixels roughly. The superiority of the *Smart-pooling* against average and max-pooling methods are shown, which are frequently used in CNNs.

The physical properties of the objects, such as weight, density, volume, and size, caused different behaviors when moving objects using a 2 degrees of freedom (DoF) robotic arm and three depth cameras in the simulation environment. For example, when the same force is applied from the same or different points to the same volume, objects that are different from one another can perform the desired movement other than tilting a light object. As a result, the physical strength of the object, as well as the strength of the force application point, emerged.

In this thesis, a test platform called *Deep Table* was created to move visually recognized objects using deep learning methods. In the *Deep Table*, there is a depth camera fixed at the top to center the robotic arm workspace, a camera in front of the workspace, a robotic mechanism that moves the variable radius pulley-variable stiffness joint system in Cartesian coordinates vertically and horizontally, micro-controllers that generate signals that drive motors with a power supply, and a computer that is used to observe sensory data collected during the experiments and to develop algorithms. We present 5 of the possible cases for the empirical test results. The results prove that our algorithm can move different types of objects successfully ranging from several grams (empty bottle) to around 250 grams (ceramic cup). The experiments also explain the role of contact point where the f/t is applied onto the object. If the contact point is adjusted conveniently, then the manipulation is terminated with a tilt over of the object. These results undoubtedly confirm that the control approach proposed in the thesis can improve the object mobility of robotic mechanisms by semantic bond extraction from visual data of objects.

GÖRSEL TANINAN NESNELERİN DERİN ÖĞRENME KULLANILARAK HAREKET ETTİRİLMESİ

ÖZET

İnsanlar, çevrelerinden en büyük ve nitelikli veriyi, görme sistemleri aracılığıyla edinirler. Bununla birlikte, daha eksiksiz ve güvenilir bir algılama için diğer duyu organlarından elde edilen verileri de tamamlayıcı bir şekilde kullanmak gerekir. İnsanlara benzer şekilde robotlar da algılayıcıları aracılığıyla içinde bulundukları ortamdan veri toplarlar. Nesne algılama, tanıma ve anlamsal değer atfetme, robotik alanındaki en güncel araştırma alanlarının başında gelmektedir. Yazılım ve donanım teknolojilerinin gelişimi akıllı sistemlerin günlük yaşama nüfuzunu sağlamaktadır. Yüksek çözünürlüklü, derinlik algılayan kameralar, nesnelerin interneti aygıtları gibi donanımlar çok boyutlu ve büyük hacimli veri elde etmeyi mümkün kılmıştır. Buna paralel olarak robotlar, endüstriyel alanda kullanımlarının yanısıra sosyal hayatın da bir parçası olarak değerlendirilmeye başlanmıştır. İnsan-robot etkileşimli sistemler içerik olarak gerçek zamanda çalışma açısından yüksek doğruluk ve hız gerektirmektedir.

Kontrolsüz artan dünya nüfusu ve dengesiz tüketimin bir sonucu olarak geleneksel üretim yöntemlerinin ihtiyaç ve talepleri karşılayamaması, zorlu rekabet şartlarının hüküm sürdüğü üretim sektöründe yeni yaklaşımları zorunlu kılmaktadır. Üretimdeki bu ihtiyaç ve zorunluluklara cevap vermek üzere Endüstri 4.0 adlı yenilikçi bir vizyon ile akıllı üretim yöntemleri ve tesisleri öne sürülmüştür. Bu vizyon kapsamında makinalar, cihazlar, sensörler ve insanlar arasında iletişimin sağlanabildiği, gerçek sistemlerin sanal fiziksel bir kopyasının dijital ortamda oluşturularak bilginin anlamsallaştırıldığı ve bilgi şeffaflığının sağlanabildiği ortamlar oluşturulması planlanmaktadır. Ayrıca insanlara zorlu şartlarda makinalar tarafından teknik destek sağlanması, siber-fiziksel sistemlerin karşılaştıkları bazı problemlerle ilgili kendi kararlarını insanlara ihtiyaç kalmadan verilebilmesi de, bu çerçevede, geleneksel üretim yöntemlerine bilişim teknolojilerinin entegrasyonu için amaçlanmaktadır.

İnsan-robot etkileşimi çerçevesinde, robotların; güvenli, hızlı ve verilen görevleri yüksek başarım/düşük hata oranlarıyla gerçekleştirebilmesi, gelişen makine öğrenmesi algoritmaları ve bu algoritmalara uygun donanım teknolojileriyle mümkün hale gelmiştir. Akıllı üretim tesislerinde robotların yer değiştirmeleri esnasındaki hareketleri, verilen görevleri beklenen performans düzeylerinde yapabilme kabiliyetleri, sahip oldukları donanım ve yazılım sistemlerine direkt olarak bağlıdır. Konvolüsyonel (evrişimsel/evrişimli) derin yapay sinir ağları daha çok nesne tanıma gibi görsel ve ses tanıma gibi ses tabanlı verilerle nesne tanıma, nesne sınırları belirleme, nesne bölütleme, anlamsal bağ oluşturma gibi amaçlarla eğitilir.

Nesnelerin interneti aygıtları ve çeşitli sensörlerden aldıkları verileri işleyerek, öğrenen sistemlerin oluşturulması modern robotik ihtiyaçlarına cevap vermektedir. Böylelikle öğrenebilen robotik mekanizmalar, işleyişleri esnasında çeşitli duyargalardan aldıkları

verileri işleyerek öğrendikleri modeller üzerinden karşılaştırma yaparak anlamsal bilgi edinimine haiz olurlar. Yenilikçi robotik yaklaşımlarda, kritik öneme sahip olan bu durum vasıtasıyla, robotlar karmaşık yapılardan anlamlı bağlar kurarak insanlara benzer davranış geliştirme özelliği kazanabilirler. Hafıza kapasiteleri, birim enerji başına performansları ve paralel hesaplamaya uygun çok çekirdekli yapılarıyla güncel grafik ekran kartları derin yapay sinir ağı yapılarının eğitilmesine ve böylece daha fazla parametre öğrenilebilen büyük boyutlu verilerin işlenmesine imkan tanımaktadır. Ayrıca gömülü sistem olarak çalışmaya uygun benzer şekilde çok çekirdekli donanımlar da gerçek zamanlı bilgisayarla görü içeren, karmaşık robotik uygulamalara imkan tanımaktadır.

Konvolüsyonel derin yapay sinir ağları, bazı özel mimari, parametre güncelleme yöntemleri ve aktivasyon fonksiyonları ile ikiden daha fazla sayıda gizli katman içeren özelleşmiş yapay sinir ağı modellerine verilen genel addır. Büyük veri kullanılarak eğitilen derin konvolüsyonel sinir ağı modelleri, eğitildikleri veri kümesinde bulunan nesneleri tanıma, nesneleri çevreleyen sınırları belirleme ve bölütleme gibi konularda insan performansından daha yüksek başarımlı, çok küçük hata değerlerine sahip sonuçlar vermektedir. Robotik alanındaki uygulamalar için nesne tanıma ve nesne sınırları belirleme işlemleri anlamsal bilgi çıkarımı ve nesnelere dayalı ilişki kurma bağlamında tek başına yetersiz kalmaktadır. Bu sebeple, nesnenin ait olduğu sınıfa, sınıf etiketlerinin ötesinde öznitelikler atanarak algoritmaların anlamsal içerik konusundan çıkarım yapabilmeleri sağlanır.

Bu tez kapsamında, geleneksel öznitelik çıkarımı yöntemlerinin algılayıcı ve açıklayıcı kısımlarının birleşimleriyle elde edilen yöntemlerin performansları detaylı bir şekilde analiz edilmiştir. Performans ölçütü olarak hız, başarım ve görüntü başına doğru eşleşen öznitelik gibi sıradan ölçütlerin yanısıra eşleşen özniteliklerin birbirlerine olan uzaklığı, doğru eşleşen öznitelik sayısı ve eşleşen noktalar arası açısal yönelim farkı gibi ölçütler de kullanılmıştır. Deneylerde, 127 adet şablon görüntüden oluşan bir sorgu veri kümesi, 3090 adet görüntüden oluşan bir veri kümesiyle eşleştirilmiştir. Bu çalışma sonucunda, bazı geleneksel öznitelik çıkarıcı yöntemlerin başarım gözönüne alındığında kabul edilebilir seviyelerde sonuçlar verdiği görülmüştür. Bununla beraber, gerçek zamanlı çalışmaya uygun herhangi bir yüksek başarımlı kombinasyon elde edilememiştir. Ayrıca geleneksel yöntemlerin getirdiği hesap yükü nedeniyle görüntü varyasyonları kısıtlı tutulmak zorundadır. Bu sonuçlar, geleneksel yöntemlerin karmaşık robotik görevlerde istenen sonuçları vermesinin mümkün olmadığını ortaya koymaktadır. Bu noktadan hareketle, tez kapsamında görsel olarak tanınan nesnelerin, robotik bir mekanizmayla hareket ettirilmesi derin öğrenme yöntemleri kullanılarak gerçekleştirilmiştir. Robotik mekanizma olarak değişken yarıçaplı makara-değişken sertlikli eklem sisteminden oluşan bir yapı, derin masa adı verdiğimiz deney ortamına entegre edilmiştir.

Robotik mekanizmalara ait çalışmalar uzun ve maliyetli deneyler gerektirmektedir. Bu ksıtların etkilerini en aza indirmek adına benzetim ortamlarından faydalanılır. Böylece zaman ve maliyetten tasarruf edilirken, birçok varyasyon denemesi yapılarak gerçek dünya deneylerine olabildiğince hazır prototiplerle başlanır. Kısıtlı kabiliyete sahip geleneksel öznitelik algılama ve tanıma yöntemlerinden ziyade konvolüsyonel sinir ağları, başarım oranı daha yüksek ve daha hızlı anlamsal bilgi elde edebilmekte, böylelikle gerçek-zamanlı robotik uygulamalara imkan tanımaktadır. Bu modellerin istenen sonuçları üretebilmeleri parametrelerinin uygun şekilde optimize edilmesine

bağlıdır. Bu da ancak yeterli sayıda veri ile mümkündür. Bu tez kapsamında robot görüsü çalışmalarında kullanılmak üzere, benzetim ortamlarıyla gerçek dünya koşulları arasındaki farkı azaltmaya yönelik, *ADORESet* adında bir görüntü veri kümesi oluşturulmuştur.

ADORESet, 30 kategoride, her bir kategoride 2500'er gerçek, 750'şer tane de benzetim ortamından alınan toplamda 97500 adet etiketli ve nesne sınırları elle işaretlenmiş görüntüden oluşmaktadır. Bu veri kümesi kullanılarak en iyi sonuçlar veren konvolüsyonel sinir ağı mimarilerinden dört tanesi ince-ayar yapılarak eğitilmiştir. Sonuç olarak ise *VGGNet* adlı algoritma nesnelerin hareket ettirilmesi esnasında tanıma işlemini yapacak yöntem olarak belirlenmiştir. Ayrıca nesne sınırlarını belirlemek için de *RetinaNet* adı verilen mimari ince-ayar yapılarak eğitilmiştir. İnce ayar yaparak eğitme işlemi, genel olarak geniş kapsamlı başarımı yüksek modelleri, ilgilenilen alanda parametreleri güncelleyerek daha başarımlı hale getirmektedir.

Konvolüsyonel sinir ağı modeli girdisi olarak sayısal piksel değerlerinden oluşan görüntü matrisi bir vektör haline dönüştürülür ve modele beslenir. Konvolüsyonel yapay sinir ağları genelde, konvolüsyon, en büyük değer havuzu, düzleştirme, normalizasyon, tamamen-bağlı gibi birçok katman içeren yapıya sahiptirler. Bu tezde, literatürdekilere alternatif bir havuzlama katmanı önerilmektedir. *Smart-pooling* adı verilen bu yöntem, ilgili filtre içerisinde ortalamadan büyük veya küçük piksel değerlerini ele alarak işlem yapmaktadır. *Smart-pooling*in, konvolüsyonel yapay sinir ağlarında en sık kullanılan, en büyük değer ve ortalama havuzlama yöntemlerine olan üstünlükleri gösterilmiştir. Görüntü girdileri bu katmanlardan, model parametreleri uygulanarak geçer ve çıktı katmanında sınıflandırılmak istenen nesneler sayısınca ünite bulunur. Bu ünitelerin her biri, farklı bir nesne sınıfını temsil etmektedir. Böylece en yüksek sayısal ünite değeri, konvolüsyonel sinir ağı girdisi olan görüntünün ait olduğu nesne sınıfını belirtmektedir. Konvolüsyonel sinir ağları geleneksel özelik algılama/tanıma yöntemlerine göre çok daha yüksek başarımlı ve gerçek zamanlı çalışmaya uygundur. Benzetim ortamında, 2 serbestlik dereceli robotik kol ve üç adet derinlik kamerası kullanılarak yapılan nesne hareket ettirme eylemlerinde, nesnelerin ağırlık, yoğunluk, hacim ve boyut gibi fiziksel özellikleri hareket etme esnasında farklı davranışlar ortaya çıkmasına neden olmuştur. Örneğin aynı ağırlık, farklı hacimdeki nesnelere aynı şiddette kuvvet, aynı veya farklı noktalardan uygulandığında hafif olan nesne devrilirken diğeri istenen hareketi gerçekleştirebilmektedir. Sonuç olarak nesne fiziksel özellikleriyle beraber kuvvet uygulama noktasının önemi ortaya çıkmıştır.

Bu tezde, görsel olarak tanınan nesnelerin derin öğrenme yöntemleri kullanılarak hareket ettirilmesi için *Deep Table* adı verilen bir deney platformu oluşturulmuştur. *Deep Table*'da, robotik kolun çalışma alanını ortalayacak şekilde tepeye sabitlenmiş bir adet derinlik kamerası, çalışma alanını karşıdan görecek şekilde bir adet kamera, kartezyen koordinatlarda değişken yarıçaplı makara-değişken sertlikli eklem sistemini dikey ve yatay eksende hareket ettiren bir robotik mekanizma, bir güç kaynağı ile motorları hareket ettiren sinyalleri üreten mikrodenetleyiciler ve deneylerde toplanan duyarga verilerini gözlemleme ve algoritma geliştirmede kullanılan bir bilgisayar bulunmaktadır. Karşıdan çalışma alanını gören kamera, nesnelere kuvvetin uygulanacağı yüksekliği hesaplamak için nesne sınırlarını ve nesne sınıfını belirlemede kullanılır. Derinlik kamerası ise nesne hareketini takip eder ve nesne hacmi hesabında kullanılacak veriyi sisteme sağlar. Bu kameralardan alınan verilerle nesne sınıflarına atanan özniteliklere göre hesaplamalar yapılır ve robotik kolun nesneye uygulayacağı

kuvvet noktası ve kuvvetin şiddeti belirlenir. Tez kapsamında olası birçok ihtimalden 5 tanesi için deney sonuçlarına yer verilmiştir. Deneylerde seramik bardağı, plastik şişe boş ve dolu durumları için hareket ettirilecek nesneler olarak kullanılmıştır. Herhangi bir akıllı kontrol yöntemi uygulanmadığında rastgele durumlar hariç nesnelerin istenen hareketi elde edilememiştir. Önerdiğimiz kontrol yaklaşımıyla, boş plastik şişe veseramik bardak gibi ağırlık, boyut ve yoğunluk gibi fiziksel özellikleri birbirinden çok farklı nesneler bile başarıyla hareket ettirilirken, aynı şişeye belli bir miktar su doldurulduktan sonra değişken yarıçaplı makara-değişken sertlikli eklem sistemi iç sertliğini en üst seviyeye çıkarsa da gerekli kuvveti sağlayamadığı için hareket belli bir noktada sonlanmıştır. Bu sonuçlar, tezde önerilen kontrol yaklaşımının nesnelere ait görsel verilerden anlamsal bağ çıkarımı ile robotik mekanizmaların nesne hareket kabiliyetlerini geliştirilebileceğini açıkça göstermektedir.

1. INTRODUCTION

Throughout the history, people have continuously searched for more advanced life conditions with explorations, inventions, findings, and optimizations they have made over what they currently have. Mental and physical evolutions have always appeared to serve this purpose as being the historical milestones. Last decades witnessed the intelligent systems to become ubiquitous in every aspect of daily life as the so-called era of information technology and the digital revolution. Even though the earlier views pointed out the weaknesses, deficiencies, imperfections, and shortcomings of machine usage within human-existing environments, recent discussions are pursued by researchers as well as wider technology communities on the replacement of human labor with robots. Due to the exponential growth of innovation in technology, machines went under a tremendous progress. Nowadays, computationally-intensive (and complex/complicated) tasks are performed by machines accurately, sometimes even better than human-levels. In other words, the success of machines in real-world applications has ceased the controversies about the human labor power substitution, but the content, size, safety, robustness, and speed of the replacement process are being debated publicly around the world. This debate is focused on big-data and information-use related areas. The latest advancements in hardware, algorithms, and software make it possible for robots to acquire semantic relations and make inferences by learning from data with deep neural networks.

The deep-seated pervasive attitude suggests that the eyes are the largest data source for humans. Once the success of deep neural networks surpassed the conventional methods in computer vision applications, prospective usage areas caused a snowball effect that made it one of the most attractive research topic. The rise of deep neural networks can be explained in the manner of an interdisciplinary progress involving computer science, robotics, manufacturing and automotive industries beyond hardware and software developments. In the context of technological progress, the famous "data is the new oil" statement allows an analogy-based definition of the current situation.

In addition, it indicates the necessity of raw data to be processed in detail to obtain valuable entities at every step similar to oil refinery which yields different products such as gas, plastic, chemicals, etc. Deep learning algorithms that make machines intelligent provide useful data from all layers as well.

1.1 Semantic Information Retrieval

In cognitive psychology, there exist various types of definitions about how humans give hereditary meaning to visually recognized objects. The study in [1] suggests that mental images, which are developed in mind without any physical support, are composed of information about the object surface and its deeper information. Upon this definition, they propose mental images can replace the actual perceptions and surface representations as being quasi-pictures that are derived from deeper information built in mind. On the contrary, [2] argues against [1], which is assumed to have deficiencies and misleads theoretical details. [2] assesses the mental imagery by introspection and it proposes that the objects in the scenes are also the properties thought of but not the memories or mind segments. In addition, the same study states that the properties of visual scenes are not coherently described in detail by anybody. It can be said that they eyes are just the sensing tools, brain infers the actions for data acquired from the eyes. Considering the earlier views, one can say that the actions taken by people about the visually perceived objects are shaped in the mind. For instance, as displayed in the Figure 1.1, the appearance in the mind for the same object or situation can be very different because of the perspective or different point of view. This difference can also occur between a healthy person and another with cognitive impairment. Therefore, beyond the exact information on the category and the location of the object is not sufficient. The crucial aim is to achieve semantic intelligence which enables the machine to answer the content, function and location of the object. Likewise, [3] devotes richer meaning to object perception than seeing it with eyes, which implies dynamic brain operations, memory queries, and inferences. Moreover, perception is defined as of predictive fact in the same study. In essence, visual perception is considered regarding its relations with brain, memory, eyes and inherited knowledge, which are analogous to decision-making systems likely to be

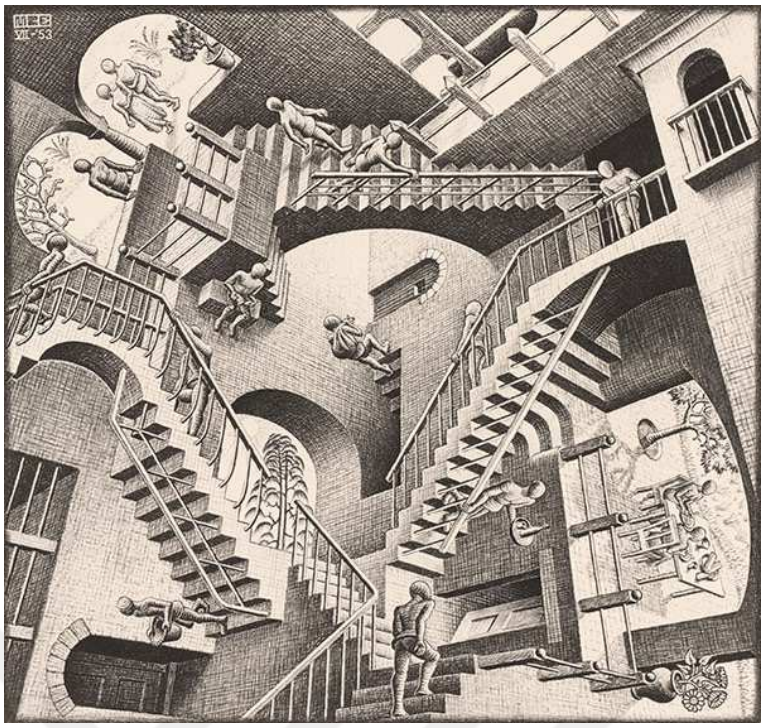


Figure 1.1 : Representative illustration to emphasize the effect of perspective for appearance of objects in the mind. [4]

algorithms, datasets and extracted features, visual perception devices and semantic representatives for objects.

Instant object recognition is an operation of calling knowledge about object identities that are stored as prior information, which is previously mapped to consistent memory segments. In computer vision, the efforts behind answering the questions of where the object of interest is in the image or what exists in the whole frame in terms of detecting and recognizing have become obsolete so that the current situation implies further endeavor to extract meaningful information from data using various approaches. Therefore, ongoing studies on machine vision systems attempt to perceive the appearance-based changes and occlusion, inter-class relations and subordinate types of same objects as stated in [5] similar to humans that it is easier to recognize basic-level classes compared to atypical objects. Based on the common behavior of of constructing the semantic relations in a hierarchical manner, (i.e. from general to particular such as organism$\longrightarrow$mammal$\longrightarrow$person$\longrightarrow$male or female), a binary classifier is introduced in [6] to recognize objects using labels to obtain information about relations between objects. In [7], a localization is performed for an infotainment robot by pose estimation using visual features within an indoor

environment. Recognized objects are utilized to distinguish the area and this semantic information specifies how the robot will interact with humans.

In the light of visual feature extraction, [8] introduces recognition based object-place relation estimator using spatial-semantic information over the assumption of a robot that is capable of recognizing real-world objects. In [9], towards the tasks requiring more comprehensive knowledge about the interaction object, a platform called KnowRob-Map is proposed. This platform consists of object spatial knowledge along with general information including category and purpose. KnowRob-Map relates the obstacles and their public meanings, which enables the robot to perform a certain task depending on the recognized objects compatible with its operating area. Another study on indoor mobile robot navigation, [10] associates recognized door signs to their semantic content encoded as text with the goal of mapping. Similarly, [11] introduces a mobile indoor robot that works in unknown environments and acquires high-level semantic features such as the room type, relationships between objects and materials of walls and grounds using a depth camera. [12] constructs a semantic objects maps including aspect and joint knowledge for kitchen furniture objects as task-relevant information, which is collected by a depth camera autonomously. The experiments of this study are conducted using a mobile robot within different kitchen environments as everyday manipulation tasks after answering some questions about the objects that are stored as lexical representations. In the same fashion, an indoor mobile robot is equipped with a depth camera to segment out objects by fusing visual data from multiple views in [13] for daily household tasks. Pixel-wise object categorization is executed utilizing a scale-invariant classifier for depth images that are placed into a 3D map semantically. The power of semantic information assignment to visual data is revealed in [14], which divides 60 semantic attributes into 5 groups as follows; *i)* scene, *ii)* color, *iii)* part, *iv)* shape, and *v)* material. The performance of the proposed exclusive classifiers towards semantic attributes together with bag-of-visual-words method outperforms the results of distinct methods. In [15], images are composed of objects in them and the spatial coordinates with appearance information are stored to object bank. They show how their object bank achieves better accuracy rates with ordinary classifiers in high-level image recognition challenges accompanied by success in semantic information retrieval.

In earlier investigations, it is remarkable that additional attributes are embedded to relevant visual features to achieve better semantic representations for high-level tasks such as content based image retrieval and search [16–20], improving robot navigation, actions and understanding [21–26], etc. mostly in a top-down hierarchical approach. Although aforementioned conventional feature extraction methods give significant results for semantic information retrieval problems, deep neural networks accomplish faster and more robust results applicable in real-time with higher performance metrics. With this in mind, [27] establishes reasonable connections between different actions by estimating sub-actions for unknown situations with the help of lexical, visual and logical tips. By training a deep neural network on 27425 web-images for consistent actions, they accomplish the problem of predicting action information from images. To succeed in dexterous robotic grasping [28] develops a deep learning approach by training its system with a multi-channel dataset, which contains 20 categories of objects, 6-axis force-torque(F/T) and tactile data for particular objects. Their experimental results reveal the importance of the amount and variety of training data but the object recognition accuracy rate has reached barely 88% due to the small number of training images.

In the past decade, a number of studies have sought to determine the importance and potential of semantic information in robotic navigation applications. [29] underlines the role of semantic information beyond geometry and appearances for map-based mobile robotic applications by consigning this problem to convolutional neural network (CNN) integrated simultaneous localization and mapping (SLAM) framework. Object recognition for 13 classes of objects is performed over real-time depth video data and the objects are localized into the 3D map. In a similar way, [30] proposes a CNN based semantic scene classification algorithm for indoor mobile robots that is intended to give meaning to scenes thereupon the recognized objects. The researchers in [31] presents a mobile humanoid robot visual system for navigation, which enables generating image dataset online during the autonomous motion to train deep neural networks. Then, the robot performs the experiments within a dynamic environment using a geometric map where the semantic map is also embedded in. As SLAM is a well-known problem that has implementations on many platforms, [32] enhances the available geometric data to achieve meaningful information by

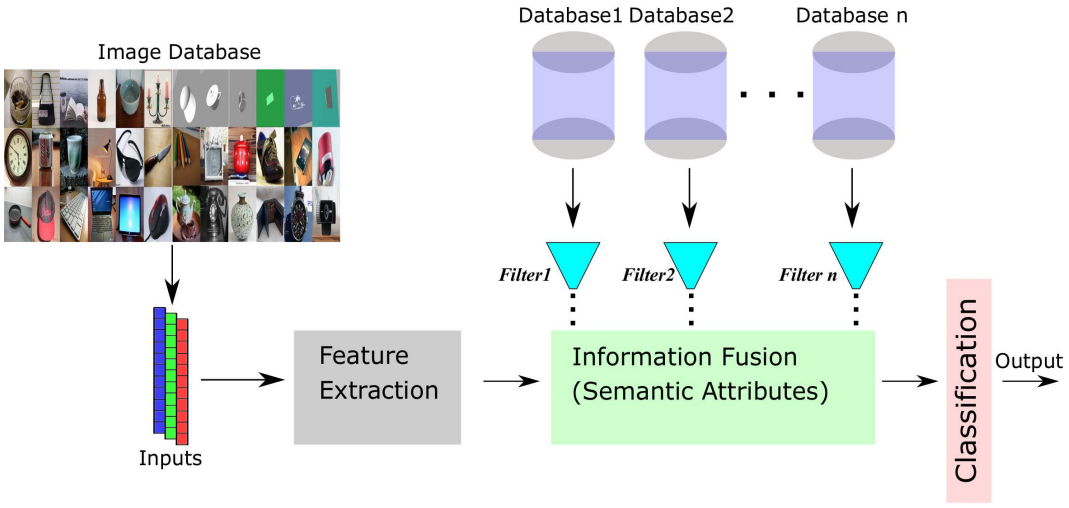


Figure 1.2 : Simplified semantic information retrieval from images.

labeling the objects inside a warehouse autonomously towards the goal of Industry 4.0 compatible system. They obtain online semantic map through the combination of SLAM with object detection and recognition using a depth camera. Similarly, [33] uses depth camera data to train a CNN model to predict semantic segmentation outputs. The multi-view geometry for object recognition is included within the system by SLAM trajectory, thus the performance of the system, which is evaluated on datasets having 13 and 40 categories, is increased. In Figure 1.2, fundamental steps are displayed for the operation of semantic information retrieval from images.

1.2 Purpose of the Thesis

In recent years, there is a big trend towards intelligent systems which evaluate the data by its size and use the so-called “deep learning (DL)” methods to process it. Moreover, this trend does not include just the algorithms, software or specific software libraries but also the hardware, especially including the graphical processing units (GPUs). Because GPUs have been improved significantly in terms of mixed precision performance, performance per watt, memory capacity, speed in terms of floating point operations per second (FLOPS), and allowing parallel computing, it has become feasible to process large-sized data. In addition to this improvement, it has been possible for DL systems to learn much more parameters than before which enables the systems to be more intelligent because larger quantity of data and better hardware are available to make dense computations. DL methods are specialized artificial

neural networks (ANNs) with more hidden layers and some specific architectures and activation functions.

This thesis aims to open up a new path in the practical application of artificial intelligence (AI) in robotics field particularly focusing on deep learning methods in the manipulation tasks of robots using visual recognition. In the general sense, the work performed here is suggesting to use the visual perception in order to achieve more intelligent manipulation actions by applying the state-of-the-art deep-learning method for decision making. While the study offers a sound methodology for solving one of the most visited problems in robotics, it also expands on the improvement of the visual recognition algorithms and deep neural networks. Moreover, a densely-labeled, high-quality image dataset, which contains real and synthetic images, towards semantic mapping, object detection, and recognition, especially for indoor robotic tasks besides simulation applications compatibility, is also introduced in this thesis.

1.3 Contribution of the Thesis

The largest amount and the richest source of data, which people gather from their environment, is obtained through the visual system. As well as people get the visual data by their eyes, their brains always work to assign meaningful information to determine the place they exist, the objects they interact or surrounded, the actions they will take, etc. using visual data. Since previously explained studies mostly depend on the encoded and ascertained information for semantic meanings to recognized objects, our study adds inferences according to the size and dimensions, and inter-class relations for recognized objects before execution of the manipulation task.

The contributions of this thesis can be summarized as follows:

(1) A comprehensive comparison of visual recognition methods is performed considering the detection/description algorithm couples of conventional feature extraction methods. This comparison has shown us and other researchers in the similar field that scale-invariant feature transform (SIFT) [34] and speeded-up robust features (SURF) [35] algorithms, when applied together gives the best recognition rate when the recognition task has to be achieved in challenging visual settings including occlusion, illumination changes, and similar challenges.

(2) A well-documented image database containing 97500 images having manually-tagged labels is formed. This dataset allows evaluating the performance of all computer vision algorithms and deep neural network algorithms. The preparation, annotations, and sorting of the dataset contribute to both computer vision and robotics field in expanding the list of available datasets.

(3) A new algorithm for deep neural network models has been proposed. The method is proven to be better in terms of accuracy rates and spatial information adaptation between layers compared to the previous algorithms.

In addition to these three tangible contributions, the methodology developed herein is applied to a real robotic-arm providing real-world performance analysis. The preparation of the test equipment and the application also provides guidelines for similar studies in adopting the newly developed AI algorithms to the practical robotics problems.

1.4 Thesis Outline

The thesis starts with the comparison of the conventional feature detector and descriptor algorithms in Chapter 2 to set the motivation in resolving the perception challenge finding the best available combination. Then, in Chapter 3, an advanced AI method named deep-learning is explored, improved and applied to the visual recognition problem. The necessity to form an original image-dataset and the content of this dataset is also reported in Chapter 3 since the data structure and the learning methods are closely tied. Next, the outputs of the visual recognition algorithm are put into use in a robotic case study having the manipulation of basic indoor objects in a push-pull task that is executed using a variable stiffness joint mechanism.

In this thesis, each chapter has a separate literature survey since the range of the topics are quite wide and the recent developments have to be reported in the right order. A single literature survey chapter would make the following of the ideas more difficult. Therefore, for the sake of clarity and order, the literature overviews are given their own space in the relevant chapter.

2. COMPARISON OF CONVENTIONAL FEATURE DETECTOR AND DESCRIPTOR ALGORITHMS

In computer vision, obtaining beneficial information from visual data has always been substantial in order to make machines automatically interpret visual data into semantic meanings. In this section, a review of feature detector and descriptor algorithms is given in conjunction with a comprehensive analysis that compares the performance of all possible combinations of these methods in terms of accuracy, speed, number of correct matches per second. Afterwards, the highlights and conclusive findings of this analysis will be given before discussing the advantages and disadvantages of conventional techniques.

2.1 Introduction

Features are the specific image parts such as corners, edges, colors, textures, etc. Computer vision algorithms search for these unique patterns, which also can be tracked and compared with each other. By systematically extracting useful material such as location and identity about 3-dimensional (3D) world from 2D images, computer vision methods help forming the machines that can see. The primary purpose of computer vision algorithms is to detect and recognize objects, no matter how complex an image frame is. As long as the appearance of objects depends on light conditions, surface reflection and vision receptor capacity, the partnerships of these properties generate textures, colors, bounds in terms of corners, edges, blobs and contours of objects or backgrounds in images as being the main elements considered by the classical feature extraction methods. Concisely, a low-level computer vision algorithm involves two parts: a distinct 2D edifice incorporating a detector and a descriptor. Intuitively, people perform visual activities by excerpting meaningful features that are localized in the region where the highest variation occur and this operation is called "feature detection". Once a feature is detected, then it can be found in other images repeatedly. Furthermore, "feature descriptions" are information sets that particularly belong to detected features. Thus, having detected features and descriptions belonging

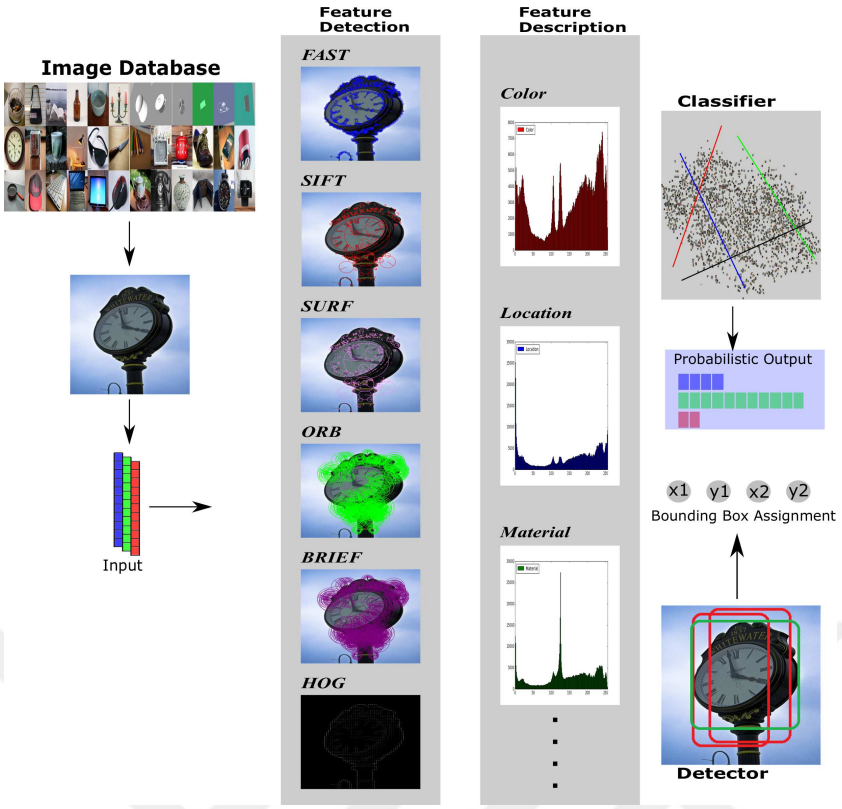


Figure 2.1 : Pipeline of a feature detector/descriptor operation for detection and classification.

to them enables to implement computer vision applications such as aligning, stitching, etc. In recognition tasks, the algorithm takes images as inputs and the outputs are usually the category labels of objects within the input images. On the other hand, the algorithm outputs pixel coordinates for the objects in detection tasks. Image datasets are required for object detection and recognition algorithms where the features are utilized to train the classifiers and regressors, respectively. Conventional feature extraction is not sufficient solely for recognition or detection so that a trainable classifier or detector has to be attached subsequently to determine the category or pixel locations as given in Figure 2.1 for computer vision applications.

A particular and influential approach for image classification is explained in [36] as a novel generic procedure. It has similar content as given pipeline with extra operations such as clustering feature descriptors by nearest neighbors, in other words, visual vocabulary construction by quantization, and binning them to histograms before Naïve Bayes or support vector machine (SVM) based classification. 1776 in-house images in 7 classes are used during the experiments.

2.2 Literature

Feature detector and descriptor algorithms imply structural or inferred information extraction about the objects in the streamed images in order to accomplish computer vision tasks. In this context, relevant semantic information retrieval from data is essential for perceiving the content of an image. Image matching, fundamentally finding the corresponding features in each compared images, is the principal operation and commencement of semantic information retrieval from numerical visual data. The outline of a conventional image matching algorithm training procedure can be summed up as follows in 5 steps:

1) Preprocessing images after customizing a relevant dataset. Cropping, shifting, resizing, color enhancing, histogram equalization, color quantization, color space transformation and spatial transformations are the most common image preprocessing techniques.

2) Extracting features from the loaded/streamed images to the system using the appropriate method those will be explained hereinafter in this section.

3) Constructing a plausible dataset composed of extracted features. The feature specifications need to be adjusted accordingly for comparison and matching purposes.

4) Training a pertinent classifier using available features and then validating the performance of the classifier until reaching a satisfactory level.

5) Comparing the features of queried images based on Euclidean distance with the ones that already exist in the dataset and decision-making about the robustness of matches, thus the class of query image is determined as a result of the comparison.

In general, the localized features are called keypoints or interest points, which indicate corners while edges are based on orientation and local appearances. Intuitively, corners are the contour junctions and are accepted as stable features over viewpoint changes. A corner detector in phases is shown in Figure 2.2. Hereby, Harris corner detection algorithm [37] can be summarized in 6 steps as follows:

1) Compute x and y derivatives of image I;

$$I_x = G_\sigma^x * I, \qquad I_y = G_\sigma^y * I \tag{2.1}$$

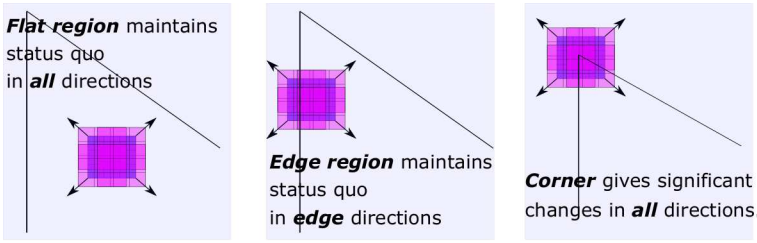


Figure 2.2 : Harris detector searching phases to detect corners.

2) Compute products of derivatives at every pixel;

$$I_{x2} = I_x.I_x, \qquad I_{y2} = I_y.I_y, \qquad I_{xy} = I_x.I_y \tag{2.2}$$

3) Compute sums of products of derivatives at each pixel;

$$S_{x2} = G_{\sigma'} * I_{x2}, \qquad S_{y2} = G_{\sigma'} * I_{y2}, \qquad S_{xy} = G_{\sigma'} * I_{xy} \tag{2.3}$$

4) Define at each pixel (x,y) the matrix;

$$H(x,y) = \begin{pmatrix} S_{x2}(x,y) & S_{xy}(x,y) \\ S_{xy}(x,y) & S_{y2}(x,y) \end{pmatrix} \tag{2.4}$$

5) Compute the response of the detector at each pixel;

$$R = Det(h) - k(Trace(H))^2 \tag{2.5}$$

6) Threshold on value of R. Compute non-max suppression.

Qualitatively, edges appear as a result of regional differences such as changes in color, intensity, and texture. Edges are described by edge normal that is the unit vector in the direction of maximum intensity change, edge direction $\theta = atan2(\frac{\partial I}{\partial y}, \frac{\partial I}{\partial x})$ that is the unit vector along edge, edge location that is the pixel coordinates where edge is located within the image, edge magnitude $\mid \nabla I \mid = \sqrt{\frac{\partial I}{\partial x}^2 + \frac{\partial I}{\partial y}^2}$ that is the local image contrast along the normal. The occurrence of an edge within an image can be computed from the gradient vector field $\nabla I = [\frac{\partial I}{\partial x}, \frac{\partial I}{\partial y}]^T$ as given in Figure 2.3. A complete edge detection algorithm [38] can be given as follows:

1) Compute x and y derivatives of image I;

$$I_x = G^x_\sigma * I, \qquad I_y = G^y_\sigma * I \tag{2.6}$$

2) Compute magnitude of gradient at every pixel;

$$M(x,y) = \mid \bigtriangledown I \mid = \sqrt{I_x^2 + I_y^2} \tag{2.7}$$

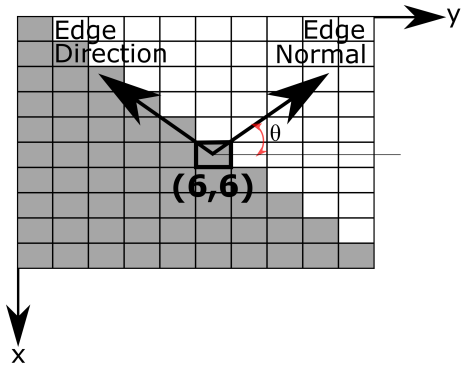


Figure 2.3 : Components of edge detection operation.

3) Eliminate those pixels that are not local maxima of the magnitude in the direction of the gradient.

4) Hysteresis thresholding;

- Select the pixels such that $M > T_h$ (high threshold)
- Collect the pixels such that $M > T_L$ (low threshold) that are neighbors of already collected edge points.

Nowadays, computations towards obtaining image features are assumed to be low-level operations but finding specific locations of the edges in images is still critical and challenging for computer vision algorithms. Since feature extraction plays a critical role in conventional image matching, SIFT [34] is assumed to be the seminal study in this territory that distil the meaningful image parts scale- and rotation-invariant manner. It is claimed in the study that distinctiveness of SIFT descriptors allow to match individual features during large database search. The scale-space representation is given by a function $L(x,y,\sigma) = G(x,y,\sigma) * I(x,y)$ that is obtained as a result of convolution$(*)$ of a variable-scale Gaussian $G(x,y,\sigma) = \frac{1}{2\pi\sigma^2}\exp^{-\frac{x^2+y^2}{\sigma^2}}$ with an input image, $I(x,y)$. SIFT applies a series of difference-of-Gaussian (DoG) filters in multiple scales, $G(x,y,s\sigma) - G(x,y,\sigma) \approx (s-1)\sigma^2\nabla^2 G)$ where s shows the scales, x and y are the spatial pixel coordinates of images. In addition, keypoint descriptor in SIFT includes a $4x4$ array containing gradient orientation histograms reciprocating the sum of gradient magnitudes. Another study, called SURF [35], goes on further direction that claims to provide as reliable and robust outputs as SIFT with faster computation and comparison times. In the detector part of SURF algorithm, interest point determinants are approximated using a Hessian matrix, which will give a

local maximum resulting in integral images. Rather than applying box filters in the decreasing direction through the image pyramid, box filters are applied directly on the original image in SURF and then the images are enlarged in the following layers with gradually bigger masks by applying filters. Non-maximum suppression in a $3 \times 3 \times 3$ region is applied to localize the keypoints. Likewise, the SIFT and SURF descriptors are 64-dimensional vectors obtained by summing Haar wavelet coefficients over 4×4 pixels around the keypoints. In the same manner, binary robust invariant scalable keypoints (BRISK) [39] method also includes feature detector and descriptor parts in itself and asserts to improve SURF by decreasing the computation time on a par with matching quality. The detector of BRISK computes a score of features from accelerated segment test (FAST) algorithm [40, 41] which is used to detect corners by evaluating the intensity of candidate pixel placed in the middle, according to the 9 consecutive pixel values of 16 pixel circle whether it is brighter or darker. BRISK descriptor, which is based on the calculations of circular sampling pattern obtained rotating by $\alpha = arctan2(g_x, g_y)$ around interest point with $N = 60$ points, comprises brightness comparison outputs as a sequence of a two-class string containing 512 bits. Another method oriented FAST and rotated BRIEF (ORB) [42] commonly utilized in favor of feature detection and description tasks, which is proposed to be an alternative to SIFT and SURF. It uses FAST as the detector and its descriptor is strengthened version of the BRIEF descriptor, which is rotation invariant. In addition, ORB goes on further arguement that it is also a faster alternative to SIFT having similar matching accuracy rates besides being robust to image noise. What is more that ORB has better descriptor performance than SURF. In ORB, the FAST detector is improved by adding *cornerness* values that are calculated by filtering Harris corner measure at each scale pyramid of images and it is also supported by an orientation component. As providing only a feature descriptor, binary robust independent elementary features (BRIEF) [43] method uses binary strings to represent keypoints. These descriptors are constructed upon pairwise intensity comparison results, which stands for image patches. It is also argued that BRIEF descriptors are not only faster than SURF or its derivatives, but also has capability to give better accuracy rates in recognition challenges if rotation invariance is absent. In either case, methods including both feature detector and descriptor parts(SIFT, SURF, BRISK, and ORB) are composed of similar operations as given in 4 steps:

1) Scale-Space Representation; constructing a scale-space, which is to upsize or downsize images, makes algorithms to achieve different image features at different octaves.

2) Keypoint Localization; LoG filter extrema give information about detected blobs such as location, size, and radius. Comparing the DoG function extrema, which are the subtraction results of images at different scales, give keypoint location candidates. Afterwards, points with low contrast are rejected and keypoints locations are already determined by refining the candidates using Taylor approximation.

3) Orientation Assignment; at every selected scale, a histogram is represented by the relevant number of bins those contain the orientation of keypoints. Orientation angles are the sum of weighted gradient magnitudes where the histogram correspondent peaks.

4) Keypoint Descriptor Operation; keypoints are described by location, orientation(rotation) and radius(scale). In fact, a feature vector containing these properties is called descriptor.

A substantial example of its kind, [44] uses the histogram of oriented gradients (HOG) as features to detect and recognize objects. Evocative to SIFT descriptors, gradient-based computations are performed and selected local extrema are transferred to histogram to determine orientation. Using overlapped local contrast normalizations their descriptor performance is increased, which is calculated on regions from dense grids. The sliding window approach is used to detect HOG descriptors. To this end, the commonly used methods are explained; however, one can also pursue to give insights about feature detector and descriptor algorithms. For further information please see [45–47].

2.3 Analysis of Feature Detector and Descriptor Methods

In computer vision applications, the differences between simulation and real-world conditions cause considerable variations in performance outputs, which directly affect the algorithms. Therefore, experiments, which are conducted by suggested techniques, have great importance. To underline the potential usage areas of conventional feature detector descriptor algorithms about semantic information retrieval in particular, a

comprehensive literature analysis is given henceforth. Even though there are blurred boundaries between studies on this subject, one can split literature into two parts; *i)* performance comparisons of detector/descriptor pairs or combinations and *ii)* analysis of detector/descriptor pairs or combinations performance outputs from a scenario based application (i.e. SLAM).

To evaluate the feature detector/descriptor combination performance results on photogrammetric applications, [48] combines 5 keypoint detectors with 2 region detectors/descriptors. The test results are assessed according to the number of correctly matched keypoints and their locations with stereo pairs for the combinations and time performance is not evaluated. With the aim of determining the best feature detector/descriptor pair, the researchers in [49] examined detector/descriptor combinations consisting of 7 detectors and 2 descriptors. The image dataset in this study contains 60 scenes from 119 position with 19 different light conditions and the area under receiver operating characteristic (ROC) curve is used to measure the performance results. In [50], which assumes SIFT descriptors as ground-truth for benchmarking, only binary descriptors and their combinations are evaluated according to the number of matched images and pixelwise distances between interest points without giving further information about total performance of different combinations and their working synergy. Beyond comparing different descriptors and keypoints for various matching techniques, [46] introduces a new descriptor as a modified version of SIFT descriptor, which is gradient location and orientation histogram (GLOH). The comparison is performed with respect to the complexity of individual parameters and usage areas with detection rate, which also includes a ROC-based evaluation criterion using recall-precision calculation for image pairs. In addition to examining the algorithm performances with regards to robustness and distinctiveness using a unified framework, [51] compares local detectors and descriptors. The proposed framework consists of 2 steps, which are calculating a detector evaluation criterion and repeatability score for 6 detectors and descriptor test for assessing distinctiveness. By the same approach, [52] modifies Harris, Hessian, and DoG detectors, thus gives accuracy and time based performance evaluations for image search and fine-grained classification tests. In like manner, [47] presents performance evaluations for affine covariant region detectors of structured and textured images by changing viewpoint

and scale with illumination and blurring variations. The outputs of the algorithm are examined with respect to repeatability for computing relevant interest points and accuracy rates for shape, scale and localization matching.

Beyond previously mentioned studies, [53] demonstrates the outcomes of feature detector/descriptor combinations tested on depth SLAM. Accuracy and running time per frame metrics are used to evaluate the results. Similarly, the experiments of [54] are performed as two different motion scenarios en route investigating visual SLAM performances of detector/descriptor pairs. Localization accuracy and motion speed of the camera for real-time performance are measured to display the effects of these pairs. In [55], depth image data is utilized to execute an autonomous quadrotor micro air vehicle motion (MAV) in an indoor environment. 3D motion of the MAV is estimated as relative motion sequence obtained in each time step by fusing the depth camera frames with inertial measurement unit (IMU) data. As an intermediate operation of visual odometry, feature extraction, where FAST is used, and matching are for determining how motion changes. Correspondingly, [56] presents a system on an air vehicle that collaborates with a ground vehicle and tracks its position by matching descriptor-free features as an alternative to global positioning system (GPS) for indoor or poor GPS coverage territories. 1D BRIEF descriptors are used to keep vertical edge properties that are extracted by applying a gradient-like filter on images captured from stereo cameras of air vehicle during tracking ground vehicle. Furthermore, [57] demonstrates the performance of feature extractors within a long-term outdoor navigation test using a mobile robot. The robustness of feature extractor combination is investigated against extrinsic variations. As a result, a trainable descriptor called generated BRIEF is proposed to tackle with seasonal weather changes and light conditions, which is comprised of a modified BRIEF descriptor using evolutionary methods. Comparatively, [58] evaluates the effects of feature detector/descriptor pair selection on visual SLAM test performance using faultless ground-truth data with respect to accuracy rates and average algorithm running times. The experiments of this study include combination results that consist of 8 detector and 7 descriptor pairs. Unlike previous studies, [59] gives real-time color feature tracking performance results of a humanoid neck system, which is given in Figure 2.4 called UMAY [60]. The neck system incorporates a set of tendon actuators in order to have the capability

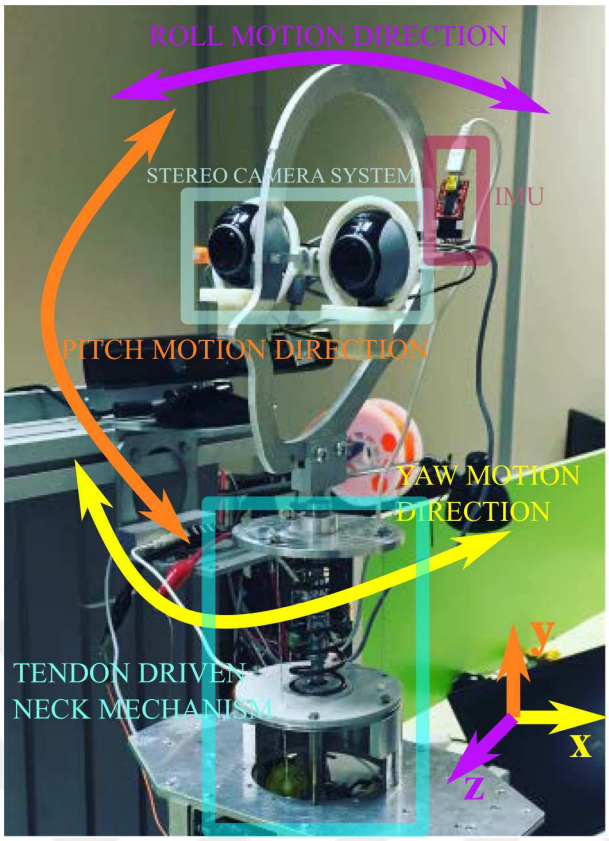


Figure 2.4 : UMAY head-neck system.

of mimicking human neck. In order to develop color tracking motion, a stereo camera system, which constitutes 2 webcams, is mounted on the head. Therefore, the algorithm is able to track color features and predefined motion profile smoothly according to cubic spline interpolation [61] after applying a hysteresis threshold to detect the specified color range. As long as the main purpose of mean shift algorithm is to find local extrema within a dataset that is used to increase the desired tracking performance in terms smoothness, and it can be expressed as follows;

- A kernel function $K(x_{i+1} - x_i$ is defined to adjust weights of the surrounding pixels of a selected pixel. In general, a Gaussian filter is applied to assign higher intensities to the pixels that are near the chosen central pixel and lower intensities to those close to the edges of the filter.

- The next weighted mean for pixels within the filter is calculated as given in Equation 2.8;

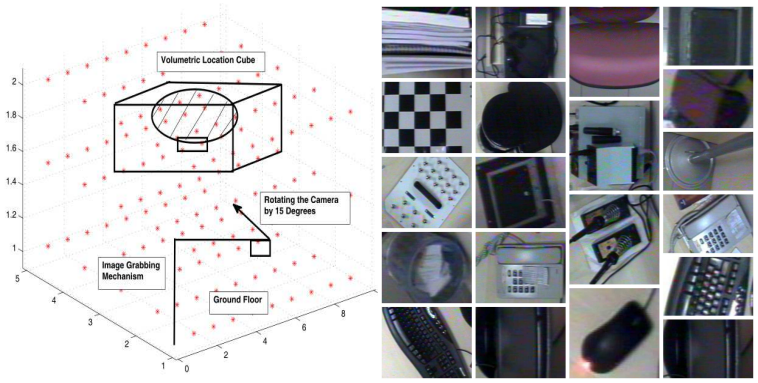
$$m(x) = \frac{\sum x_i \in N(x) K(x_i - x) x_i}{\sum x_i \in N(x) K(x_i - x)}, \tag{2.8}$$

where $N(x)$ denotes the dataset that contains the surrounding pixels around center x for which $K(x) \neq 0$

- The mean shift vector is the distance, which the filter has to move by to reach the pixel with maximum intensity. Mean shift vector can be obtained by $M = m(x) - x$
- These steps are repeated until the distance becomes zero or a predefined error rate near zero $M \cong 0$, as the condition that the filter center converges to the local extrema.

In the context of broadening valuable information about feature detector and descriptor method performance results obtained from convenient experiments, [45] is one of the most inclusive studies that investigates the feature detector and descriptor combination performances in terms of accuracy, speed, and robustness. The experiments within a blimp localization scenario conducted during this study use 555×480-pixel images for 23 feature detector/descriptor performance evaluation results regarding accuracy and speed. Indoor images (The dataset can be publicly available at: `https://web.itu.edu.tr/bayraktare/Visual_Indoor_Dataset.rar`) are gathered within an indoor office environment. The dataset consists of 3090 images which we collected from 45 points at 3 height levels and after grabbing each image the mechanism is rotated by 15° counter-clockwise as illustrated in the left side of the Figure 2.5 a). Thus, the proposed algorithm is able to locate the blimp in a volumetric cube as a result of image matching. 127 different images inside the office without occlusions are specified as to be template image, which will be queried to the algorithm. Some of these template images are given at the right side of the Figure 2.5 a), where the location information is also assigned to. Images belong to 2 points at 3 height levels are shown in Figure 2.5 b). Moreover, 19 feature detector/descriptor combinations are examined to measure robustness by evaluating the number of correct matches, mean angle difference between keypoints, and minimum distance metrics. The localization experiments are performed for a limited trajectory composed of a path with 560 images.

In brief, there are various approaches towards investigating the performance of conventional feature detector and descriptor pairs. These studies are performed in the direction of assessing various parameters affecting onto the feature detector/descriptor combinations performance outputs when they are used within SLAM, localization



(a) Schematic representation of image acquisition from office with volumetric location cube (red dots show the position of the places where images are grabbed) and examples from template images

(b) Example images from two points (red dots above) with 3 height levels

Figure 2.5 : Image-grabbing process and the hypothetical location cube, template images and a collage of query images grabbed from two points given in [45]

or tracking scenarios. To analyze the overall outcomes acquired experimentally; accuracy, distance, energy consumption, robustness, computational cost, time. Moreover, these results are tested onto maintain equivalent performances regarding repeatability under different software based or physical differences that affect the feature detector/descriptor combinations outcomes.

2.4 Results

Feature detector/descriptor based computer vision applications have a decreasing trend in recent years. Nevertheless, there are many studies in literature in the direction of

feature detector/descriptor combinations based applications such as object detection and recognition, semantic mapping, visual SLAM, etc. On the other hand, it is still unclear that which combination is the best choice for a specific task or what fundamental parameters are affecting onto which performance outcomes. The main idea behind feature based applications is to correctly match the features as fast as possible to be appropriately implemented in real-time. In addition, robustness and repeatability are essential properties that have to be taken into account for reliable and reproducible algorithm results. Therefore, this section demonstrates the remarkable performance results from aforementioned feature detector/descriptor based studies.

As a result of the tests, which are rotating images for 45°, scaling with a factor of 2.5 and rotating 45° together, applying affine transformation by changing the viewpoint of the camera for 60°, and changing the light conditions, [46] states that SIFT descriptor yield better results except for illumination changes. Steerable filters that are computed on image patches normalized to affine photometric and geometric transformations are the runner-up method, which is the best for handling different light conditions. [47] observes the decline in performance of 6 detectors depending on the increase of viewpoint change. Overlap error and repeatability are defined as theoretical metrics for accuracy. Thus, the region distinctiveness metric in terms of matching performance by looking the number of correctly matched keypoints are defined with respect to disturbances such as viewpoint, light and scale changes, blurring and compression artifacts. With the assumption of SIFT providing the ground-truth, it is argued that maximally stable extremal regions (MSER) [62] obtains better results if images include homogeneous regions with distinctive boundaries; however Hessian-Affine and Harris-Affine detectors give more regions, which is noted as a desired property that allow matching objects in the case of occlusion or clutter. In [49], the performance evaluation is composed of the results acquired from the combinations of 4 feature descriptors and 7 feature detectors. Dataset consists of images belong to 60 scenes those are grabbed from 119 different viewpoints on arc shaped paths at 3 height levels. Table 2.1 displays the area under the ROC curve (AUC) for 28 combinations. AUC is a metric, which shows the detector/descriptor combination performance as a result of matching an image pair. It is stated that DoG or MSER show the best performance regarding the detector types albeit SIFT or DAISY [63–66] is the descriptor. These

Table 2.1 : Mean area under ROC curve for 28 combinations. [49]

Detectors	Cross-Corr.	SIFT	DAISY I	DAISY II	Avg.
Harris	0.615	0.767	0.729	0.741	0.713
Harris-Affine	0.629	0.818	0.791	0.798	0.759
Harris-Laplace	0.635	0.814	0.784	0.790	0.756
Hessian-Affine	0.636	0.795	0.773	0.779	0.746
Hessian-Laplace	0.630	0.757	0.740	0.742	0.717
MSER	0.648	0.846	0.826	0.832	0.788
DoG	0.646	0.849	0.837	0.843	0.794
Avg.	0.634	0.807	0.783	0.789	0.753

4 combinations result in very close AUC scores. It is also added that in the case of small changes in scale, Harris corner detector can be preferable in terms of speed and easiness of implementing.

[50] performs on a dataset composed of 8 categories 6 images per category and 6 distortions are applied to the images that are blurring, rotating, scale changing, compression with JPEG, illumination changing. Five of the images in each category are distorted strongly while one of them remains as original. All distorted images are compared with the original ones provided that being in the same category. Having the aim of performance measurement for different methods, SIFT is assumed to be the baseline method. The outputs show that ORB detector with FREAK [67] descriptor is better in terms of accuracy. On the other hand, FAST detector with BRIEF descriptor is the fastest combination given in this study. In [51], performance evaluations are investigated for 6 detectors and 6 descriptors. Detector performances are measured according to repeatability as a identical to localization accuracy and interest point estimation under disturbances. Meanwhile, descriptor performances are evaluated by comparing ROC detection rate, which is the correct matches number over corresponding regions, with false positive rate. A match count is correct if the distance between descriptors is smaller than a threshold value and it is also verified with a ground-truth homography. There are 1000 images in the dataset, which gradual viewpoint and scale changes, blurring, JPEG compression and light variation are applied to. Conducted experiment results reveal that MSER achieves best repeatability and accuracy scores when the regions are extracted by Harris-Affine

and Hessian-Affine detectors as long as SIFT is observed as the superior method as a consequence of descriptor evaluations.

Feature detector/descriptor performances in terms of accuracy and time on visual SLAM applications are crucial due to the necessity of detected interest points locations within the environment by SLAM algorithm and assigning meaningful informations to detected keypoints. With this intention, [53] uses a dataset that is composed of colored depth images to analyse the visual SLAM performances of SIFT, SURF, BRIEF, ORB, BRISK and FREAK by measuring the difference between the ground-truth and the estimated trajectory with absolute trajectory error (ATE) metric. In consideration of accuracy and time together, STAR [68]/FREAK combination obtains the best scores with 3.79 cm root mean square of ATE and 0.1657 s to process each frame in average. However, the fastest combination is ORB/FREAK with 0.0866 s for each frame on average and SIFT with SURF based combinations give the most accurate results. For the purpose of comparing descriptor performances on visual SLAM applications, [54] uses two datasets to evaluate SIFT, SURF, BRIEF, ORB, BRISK and FREAK combinations. It is stated that SIFT gives the best results even in real-time for the both datasets. It is claimed that the descriptor performance does not affect on the visual SLAM accuracy, but it is important for real-time performance. The study suggests that SIFT extract features than SURF and a combination of SIFT with BRIEF can yield in more desirable results. In view of assessing feature detector/desciptor pairs performance on visual SLAM [58] examines results with regard to root-mean-squared-error (RMSE) for detector-descriptor combinations, maximum ATE for each reconstructed trajectory, and the average processing time. FAST detector and SIFT descriptor combination is claimed to be the best method w.r.t. RMSE score as well as FAST gives the fastest results with either BRIEF or ORB descriptors. Although it is argued that template matching is slower, FAST/SIFT combination is 4 times slower than it. In addition, FAST detector and BRIEF descriptor pair is the fastest and second in accuracy score.

Beyond preceding studies, in [45], we compare the feature detector/descriptor combination performances w.r.t. time for total computation, accuracy, and number of correct matches per second, which are given in Table 2.2 and mean keypoint angle differences, number of correct matches and distance metrics of matches, which are

Table 2.2 : Metrics for performance analysis for feature detector and descriptor pairs.

Detector	Descriptor	Time	Acc[%]	Ground-Truth	Correct Match/[s]
ORB	BRIEF	21303.30	62.83	$127 \times 3 \times 5$	1457.01
	BRISK	23461.68	74.28	$127 \times 3 \times 5$	956.16
	SIFT	97603.70	72.28	$127 \times 3 \times 5$	318.01
	SURF	79391.06	97.90	$127 \times 3 \times 5$	390.97
	ORB	21330.02	63.62	$127 \times 3 \times 5$	1455.19
SURF	BRIEF	32277.70	62.36	$127 \times 3 \times 5$	1568.79
	BRISK	35133.18	63.52	$127 \times 3 \times 5$	1275.43
	SIFT	196487.67	68.82	$127 \times 3 \times 5$	280.43
	SURF	79135.06	89.54	$127 \times 3 \times 5$	696.30
	ORB	35074.99	63.78	$127 \times 3 \times 5$	1414.50
SIFT	BRIEF	30938.27	62.73	$127 \times 3 \times 5$	879.92
	BRISK	32422.06	64.67	$127 \times 3 \times 5$	920.44
	SIFT	45919.05	62.31	$127 \times 3 \times 5$	698.46
	SURF	35319.08	98.41	$127 \times 3 \times 5$	908.02
FAST	BRIEF	23355.70	62.52	$127 \times 3 \times 5$	2736.88
	BRISK	33752.85	63.20	$127 \times 3 \times 5$	454.34
	SIFT	56152.36	72.44	$127 \times 3 \times 5$	1373.78
	SURF	37357.53	88.30	$127 \times 3 \times 5$	2065.00
	ORB	22734.26	62.62	$127 \times 3 \times 5$	275.13
BRISK	BRIEF	20517.53	64.62	$127 \times 3 \times 5$	320.69
	SIFT	34284.95	86.61	$127 \times 3 \times 5$	202.39
	SURF	26043.99	80.32	$127 \times 3 \times 5$	266.44
	ORB	23335.77	69.76	$127 \times 3 \times 5$	289.84

shown in Figure 2.6. The 'Time' column shows the total algorithm running time from start to end for the relevant feature detector/descriptor pair executed the same path to compare 560 query images with 127 template images for a total of 71120 images. As one can infer that all of the combinations are able to correctly match template images, which are selected from the dataset, it is trivial to calculate accuracy by $Accuracy[\%] = \frac{\sum(TruePositives+TrueNegatives)}{\sum TotalCases} \times 100$. Hence, we formulated a more reasonable formula to calculate a more reliable accuracy rate that takes the images into account from 3 height levels and 5 of the side images of the current position that are equally separated orientations from $[-30°, 30°]$. Hence, we have the ground-truth images by multiplying 127 template images with 3 heights for 5 side positions. We specified a histogram threshold value as 0.9 to eliminate the template images from the comparison operation from the accuracy calculations (i.e. if histogram similarity is over this threshold as a result of the comparison, then the comparison is assumed to be performed for the same images). It is clear from Table 2.2 that ORB is the fastest

method for both detector and descriptor. At the same time, the minimum number of correct matches per second is obtained by BRISK/SIFT pair. In contrast, the maximum number of correct matches per second belongs FAST/BRIEF. Moreover, SURF/SIFT pair gets the longest running time.

In Figure 2.6, the performance results for different orientations by matching the query and template images within a rotational pose range of $[-30°, 30°]$ for 5 cases are shown. These performance results are evaluated w.r.t. the number of correct matches, minimum distance metrics, and the average of angle difference values between keypoints. The lower the average of angle difference values between keypoints and minimum distance metrics are, the higher the number of correct matches for the best result. In summary, for all sub-figures displayed, the lines emphasizing the distances from the central point, which represents the best method, indicate the relative performance of each method. Thus, as the distance grows greater, the performance becomes worse. FAST/SURF pair obtains the best results for all rotations for the negative rotation cases. If the minimum distance and the number of correct matches are aimed to be optimized for a particular application, then for $15°$ and $30°$ one can use 4 of the following combinations appropriately: SURF/SURF, SURF/BRIEF, FAST/ORB, and SURF/BRISK. It is possible to make such an informed decision for any kind of priorities from the given performance evaluation parameters. For the same location case shown in the last row, SURF/SURF and SURF/SIFT pairs give the maximum number of correct matches, and SIFT/BRIEF, ORB/BRISK, SIFT/SURF, SIFT/BRISK, and SIFT/SIFT are very close to each other to give the minimum number of correct matches.

2.5 Discussion and Conclusion

An extensive analysis of visual feature detectors and descriptors is given for existing detectors and descriptors separately with their performance outputs. It is still a prominent necessity to develop successful feature detectors and descriptors one-by-one to make visual systems capable of extracting semantic information solely from visual data. Regardless from the progress on hardware, robotic applications, especially the ones that should run onboard and/or embedded hardware as well as including autonomous systems such as localization of mobile and aerial vehicles,

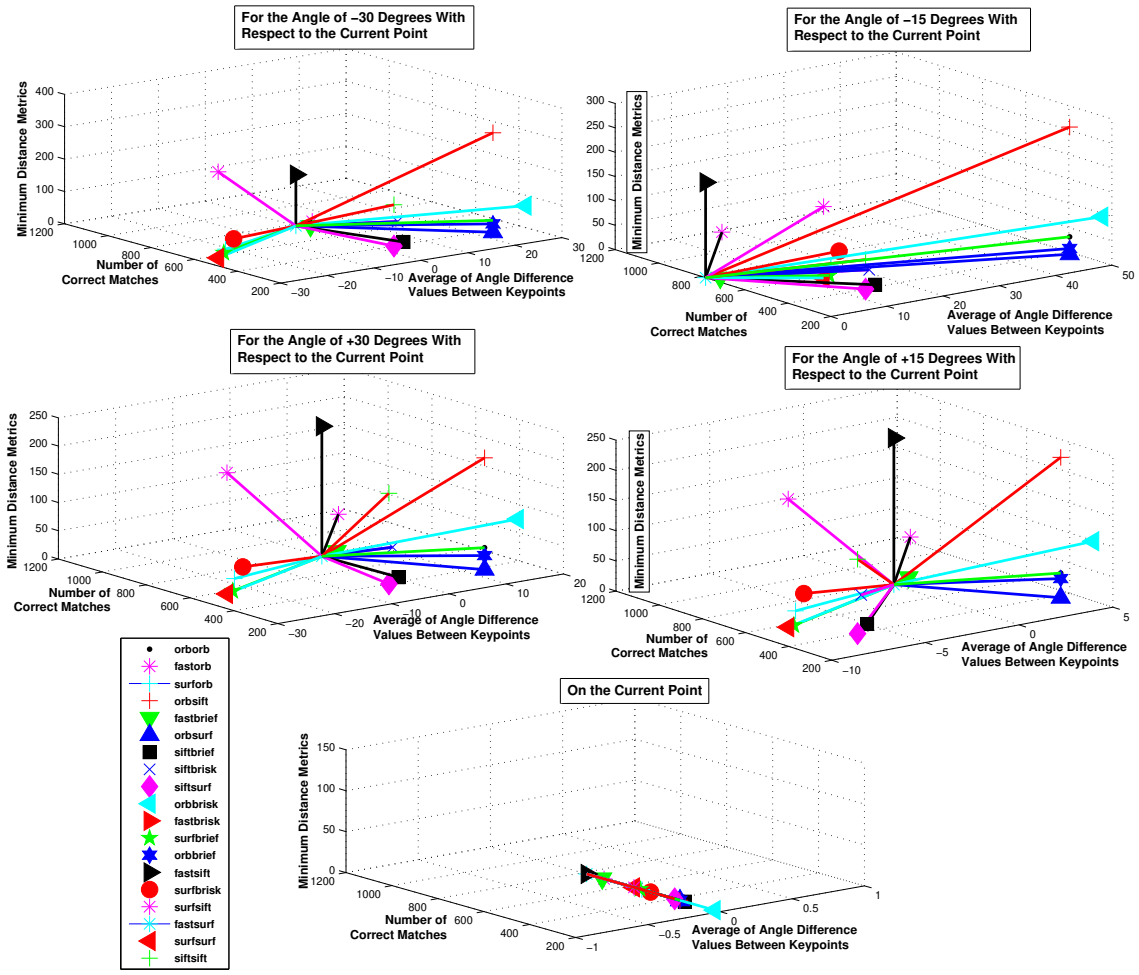


Figure 2.6 : Performance results for negative rotation at the first row while positive rotations at the second row, the current point at the last row.

SLAM, manipulation by robotic arms, etc. require improvement of existing feature detector/descriptor performance metrics in the context of computation time, robustness, reliability, repeatability, energy efficiency, accuracy, and temporal costs. The trade-offs between conditional parameters and performance results are significant, for instance, one has to compromise from robustness, accuracy, and reliability if the speed and computational costs are priorities. Our study constructs a generic structure that allows choosing the best option from the given combination judiciously for the specific priorities, besides providing a conceptual framework to extract semantic information. The application studies discussed in this study excluding the ones of proposing new feature detector or descriptor methods come up with suggestions to provide the best experimental platform or metrics to measure the performance of feature detector/descriptor pairs; however, there is not a consensus on a combination that is superior to others by surpassing in all or even many of the performance measurements.

As long as processing visual data and extracting semantic information is imperative for computer vision and robotic applications in terms of object detection and recognition along with performing the given tasks by interpreting the provided data, the conventional feature detector and descriptor methods remain giving unsatisfactory results. Therefore, deep neural network architectures, which will be detailed in the next chapter, exceed the performance of conventional feature detector and descriptor methods w.r.t. the analyzed performance metrics. They are proposed to overcome the deficiencies of these methods.

3. DEEP NEURAL NETWORKS

Technological progress paved the path of intelligent systems to penetrate the most of daily routines and this enforced researchers to evaluate the opportunities including visual systems and robots for plunging into everyday life more efficiently and naturally. In this chapter, beyond introducing a novel image dataset, deep learning architectures towards computer vision applications are explained in detail. In addition, a new pooling layer is proposed with the aim of improving deep neural network outputs. Since giving detailed explanations and processes of neural network components and operations are not the primary scope of this thesis, we will just provide an ample amount of information consistent with the context of the thesis. We will conclude this chapter by the performance results for the proposed pooling layer, fine-tuning procedures for existing CNN models suitable for our new dataset with the goal of detecting and recognizing objects inside the images.

3.1 Introduction

The contemporary approach focuses on machine learning methods to build intelligent systems. Together with the smart products as their results, these methods are generating considerable interest having great potential to cope with unsolvable problems by conventional techniques. Generally agreed definition upon machine learning concept is made by [69] as follows; *"A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E"*. Namely, as [70] explains the components of machine learning algorithms, which are the tasks that are the ways to process examples, the performance measurements that are the quantitative metrics for evaluating the abilities of the machine learning algorithm, and the experience that is the data in a manner of supervised or unsupervised fashion according to the learning style. Reinforcement learning is a kind of mixture method of supervised and unsupervised learning, which targets to infer optimal actions

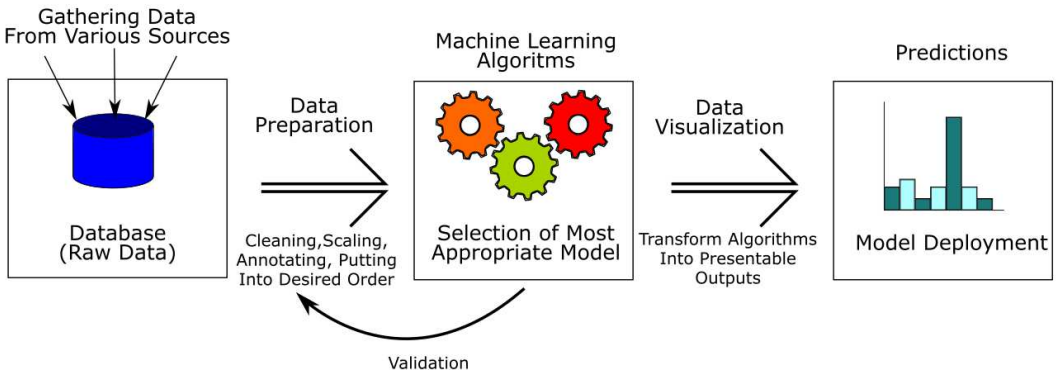


Figure 3.1 : The regular machine learning system workflow.

according to the reward scores received as a result of previous efforts. In like manner, evolutionary learning that includes methods those inspire from biological evolution can be shown as another type of machine learning. Machine learning workflow can be displayed as given in Figure 3.1. Regularly, collecting useful data for the application is the initial step for a machine learning system. Then, the received data is prepared for the determined machine learning algorithm by preprocessing with cleaning, scaling, annotating, etc. Afterwards, the data is fed to the machine learning model, and the outputs are evaluated with the validation data, which is already split from the training data. In the final step when desired performance rates are achieved, the outputs are transformed into the plausible formats for data visualization and model deployment.

In spite of there are many machine learning tasks, classification, regression, machine translation, anomaly detection, clustering, synthesis and sampling, denoising, and probability distribution estimation can be given as the most common tasks. The performance measure is a kind of quantitative metric that is specific to the task being performed by the algorithm. Accuracy is one of the most used metric evaluating the classification performance, and the regression performance is measured with the error rate, whilst the average log-probability the algorithm assigns to samples for measuring the continuous-valued score for each sample evaluating the task performance. One can concisely explain the supervised learning as experiencing a dataset of examples each of which are labeled or associated with a target while labels or target do not exist in unsupervised learning. In supervised learning, the learning process is performed over training the algorithm for learning to predict labels or target values $\mathbf{y}$ from random exemplars $\mathbf{x}$, mostly the estimation is done by $p(\mathbf{y}|\mathbf{x})$. On the contrary, unsupervised learning aims to learn a probability distribution $p(\mathbf{y})$ over given random exemplar $\mathbf{x}$.

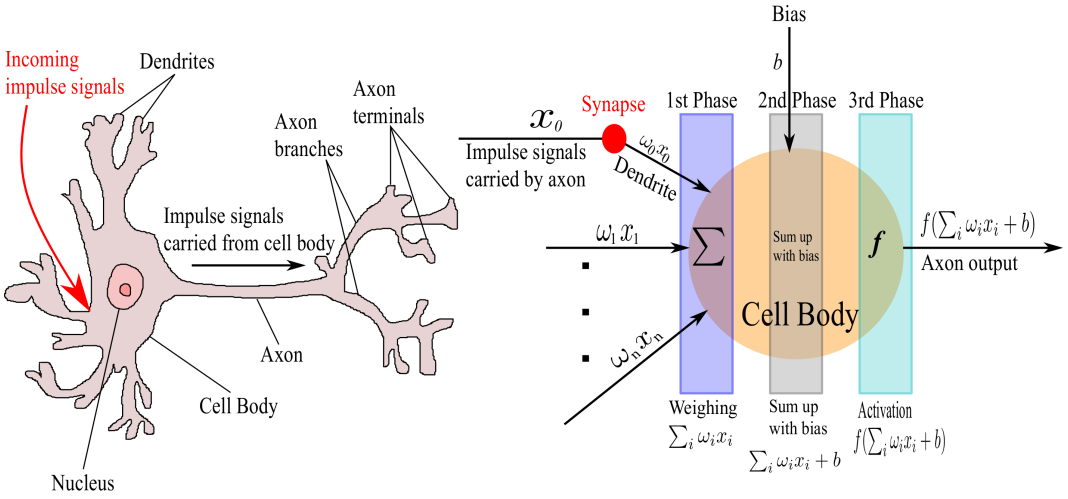


Figure 3.2 : Mathematical representation of a perceptron analogous to the biological neuron cell.

In this thesis, we focus on object detection in terms of a regression problem estimating the bounding-box pixel coordinates of the objects and object recognition regarding a classification problem estimating the labels of objects as unsupervised learning. In accordance with these aims deep neural networks (DNNs) are used, which are privatized artificial neural networks (ANNs) with more hidden layers and some specific architectures.

Theoretically, ANNs are assumed to be global function approximators, whereas perceptron is the building-block of these networks. The analogy between a biological neuron model and its mathematical interpretation perceptron are given in Figure 3.2. It is apparent that perceptron is a model that is developed by inspiration from a biological model. At the first phase of the perceptron, the input signals $\mathbf{x}$ comes as a vector containing inputs from the dataset $\mathbf{x} = [x_0 \quad x_1 \quad x_2 \quad \dots \quad x_n]$ and these inputs are multiplied by appropriate weight values $\omega = [\omega_0 \quad \omega_1 \quad \omega_2 \quad \dots \quad \omega_n]$, the bias b is added up to the weighed inputs at the second phase, and at the third phase activation the function is applied to the bias added weighed inputs, which yield the output of the perceptron as a function of $y = f(\mathbf{x}, \omega) = f(\sum_i(\omega_i x_i + b)$. Using perceptrons, one can design different types of function approximators, and these models are typically called multi-layer perceptrons (MLPs), which are the same as ANNs.

It is universal philosophy that the objective of training ANNs is to learn the optimum weights or parameters. The learning process can be divided into two parts; *i)* computing the outputs using the given inputs and current weights, and *ii)* updating

the weights according to the error. The first step is called forward-propagation and the second step is called backpropagation. We explained the operation of output calculation of a perceptron. A relevant dataset, a cost function that is also known as loss or objective function, an optimization strategy and a model constitutes the fundamental characteristics of deep learning methods. The optimization procedure, namely backpropagation, of deep neural networks is performed by sending the difference between the outputs and the targets backward through the network. Therefore, information flow is preserved in both directions. As it is alleged in [70] that training the ANNs is predominantly performed based on descending the cost function value in either way using the gradient.

One of the major issues machine learning methods encounter with is overfitting. To overcome this issue, the cost functions are penalized by adding a regularization term. In many cases, $L1-norm$, which is also known as absolute errors, $L2-norm$, which is also known as least squares error, and *dropout* [71] are used as regularization terms. L1-norm simply minimizes the absolute differences between the outputs and targets $L1 = \sum_{i=1}^{n} \mid y_i - f(x_i) \mid$ and $L2-norm$ basically minimizes the sum of the square of the differences the outputs and targets $L2 = \sum_{i=1}^{n} (y_i - f(x_i))^2$. On the other hand, *dropout* is a technique that trains sub-models for each training sample after computing the probabilistic gradient of sub-model inputs and outputs, then simply the full model is performed by dividing the weights by 2. The form of the cost function varies according to the network model and relies upon the probability distribution. Recently, ANNs principally trained using maximum likelihood, which are seeking to classify inputs. For these type of training cross-entropy based cost function is used as follows;

$$J(\omega) = \frac{1}{2} E_{x,y\sim\hat{p}} \mid\mid (y - f(x;\omega) \mid\mid^2 + Constant \tag{3.1}$$

where E denotes the expectation w.r.t. the empirical probability distribution $\hat{p}$, ω shows the parameters and x and y are the inputs and labels, respectively. Equivalently the following cost function can be written;

$$J(\omega) = \frac{-1}{m} [\sum_{i=1}^{m} y^{(i)} log h_\omega(x)^{(i)} + (1 - y^{(i)}) log(1 - h_\omega(x)^{(i)})] \tag{3.2}$$

where m is the number of samples and h_ω stands for the predicted outputs. Additionally, ANNs that are trying to solve regression problems use the following

form as cost function;

$$J(\omega) = \frac{1}{2m}[\sum_{i=1}^{m}(h_\omega(x)^{(i)} - y^{(i)})^2 + \lambda \sum_{j=1}^{m} \omega_j^2] \tag{3.3}$$

where λ is the regularization coefficient.

Suppose that a cost function for a neural network with a regularization term is given as in Equation 3.4, then one can follow the steps given in Algorithm 1 to train the ANN model by backpropagating the error until a stopping criterion is satisfied such as reaching the predetermined number of back and forth passes through neurons for updating weights or achieving a desired ε value as a result of gradient checking.

$$J(\omega) = -\frac{1}{m}\sum_{i=1}^{m}\sum_{k=1}^{K}(y_k^{(i)} \log\left(h_\omega(x^i)\right)_k + (1-y_k^{(i)}) \log\left(1-h_\omega(x^i)\right)) + \frac{\lambda}{2m}\sum_{l=1}^{L}\sum_{i=1}^{s_l}\sum_{j=1}^{sl+1}(\omega_{ji}^{(l)})^2 \tag{3.4}$$

Algorithm 1 How backpropagation works to minimize the cost function.

Require: Training dataset with m samples; Dataset: $\{(x_1,y_1),(x_2,y_2),...,(x_m,y_m)\}$

Ensure: $\varepsilon < \dfrac{\delta}{\delta\omega_{ji}^{(l)}}J(\omega)$

i) Initialization with randomly selected weights that are close to zero

ii) Implement forward propagation to get $h_\omega(x^i)$ for any (x^i)

iii) Implement code to compute $J(\omega)$

iv) Implement backpropagation to compute partial derivatives $\dfrac{\delta}{\delta\omega_{ji}^{(}l)}J(\omega)$

for $i = 1 : m$ **do**

a) Perform forward and backpropagation using sample $x^(i), y^(i)$ (get activation outputs, $a^{(l)}$, and delta terms, $\delta^{(l)}$ for $l = 2,3,...,L$)

b) Compute $\dfrac{\delta}{\delta\omega_{ji}^{(l)}}J(\omega)$

end for

v) Use gradient checking to compare $\dfrac{\delta}{\delta\omega_{ji}^{(l)}}J(\omega)$ computed using backpropagation vs. using numerical estimate of gradient of $J(\omega)$. Then disable gradient checking.

vi) Try to minimize $J(\omega)$ w.r.t. ω

It is previously said that backpropagation is implemented via mostly a gradient-based optimization method. There also exists evolutionary concepts alternative to backpropagation; however, these methods are not in the scope of this thesis. The fundamental gradient descent pseudo-code is given as follows in Algorithm 2:

Algorithm 2 Principal steps of gradient-descent method.

Require: Initial weights vector; $\omega = [\omega_0, \quad \omega_1, \quad \omega_2, \quad ..., \quad \omega_m]$

Ensure: Determine a small enough learning rate; η

Repeat until reaching a satisfactory ε rate;

$\omega \leftarrow \omega - \eta \frac{\delta}{\delta \omega} J(\omega)$

Using provided knowledge about even deep neural networks can be trained rather than shallow networks. The necessary information about CNNs with the purpose of object detection and recognition tasks will be expressed in the next sections of this chapter.

3.2 Literature

For the reason, a complete section commits an extensive material about image datasets in this thesis; this section considers the literature concerning the prominent deep convolutional neural networks with a concentration on object recognition and object detection. With the advancement of datasets and the progress of computational skills of hardware, GPUs, in particular, eased to process numerous images within performance objectives. Concisely, it is theoretical path for deep learning research to concentrate on a condensed problem, gather unprocessed data and clean it, then ameliorate existing methods by fine-tuning or deriving depending on data to attain reasonable results.

Early studies about deep learning techniques and their applications are reviewed in [72] which presents a broader insight predominantly on the theme of CNNs considering object detection and recognition. Besides, it gives some details about recurrent neural networks (RNNs) and its usage areas are mainly in text processing. It suggests that CNNs are more proper for the image, video, speech, audio processing applications, and RNNs are for text and speech processing. The basic convolution operation in one dimension between two functions is shown in Equation 3.5. Although it is useful for signal processing and time series prediction, 2D and 3D convolutional operators are required to process higher dimensional data, for example, images and videos.

$$[f * g](t) = \int_{-\infty}^{\infty} f(\tau) g(t-\tau) d\tau = \int_{-\infty}^{\infty} g(\tau) f(t-\tau) d\tau \tag{3.5}$$

where $*$ stands for the convolution operation.

Despite the fact that the CNN architecture called LeNet [73] can be claimed to be the very first and successful CNN model applied to optical character recognition (OCR)

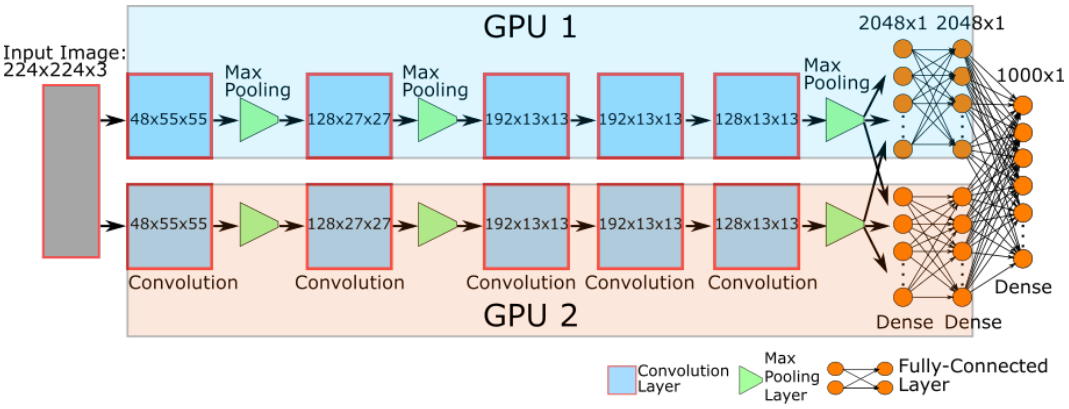


Figure 3.3 : The *AlexNet* CNN architecture.

tasks widely in 1998, researches on CNNs did not follow the path after it because of hardware, software, and data issues. LeNet is trained on handwriting images dataset [74], which is composed of 70000 handwritten digits, 60000 for training and 10000 for validation with the dimension of 28×28 in pixels. The CNN model achieves an accuracy rate of 99.2% on test session with its 5 hidden layers, which are 2 convolutional layers each accompanied by a pooling layer and a fully connected (FC) layer before classification layer. Deep Learning methods had been around for a long-term, but they turned into the mainstream in computer vision with its echoing advance at the ImageNet Large Scale Visual Recognition Challenge (ILSVRC) [75] of 2012. In that competition, an algorithm based on CNNs [76] called *AlexNet* shook the computer vision field with a stunning accuracy that is better than the method that picked up the second rank as being the solely CNN based registration. As can be assumed to a modified version of the *LeNet* model with max-pooling and rectified linear unit (ReLU) activation function besides *dropout* regularization. The *AlexNet* architecture has 7 hidden layers, 5 of them are convolutional layers, where the first, second, and fifth are followed by max-pooling, and the rest are fully connected (FC) layers before softmax classification layer, which keep 60 million parameters between 650000 neurons. It is clear from Figure 3.3 showing the AlexNet architecture that the CNN model is sliced into two symmetric parts, the reason beneath is the usage of two GPUs in parallel to train the network with the goal of reducing training time. The training images, which are 1.2 million images in 1000 classes, are rescaled for *AlexNet* to the dimension of $224 \times 224 \times 3$ and the nonlinearity is maintained by ReLU. The *dropout* method is against reducing overfitting issues in

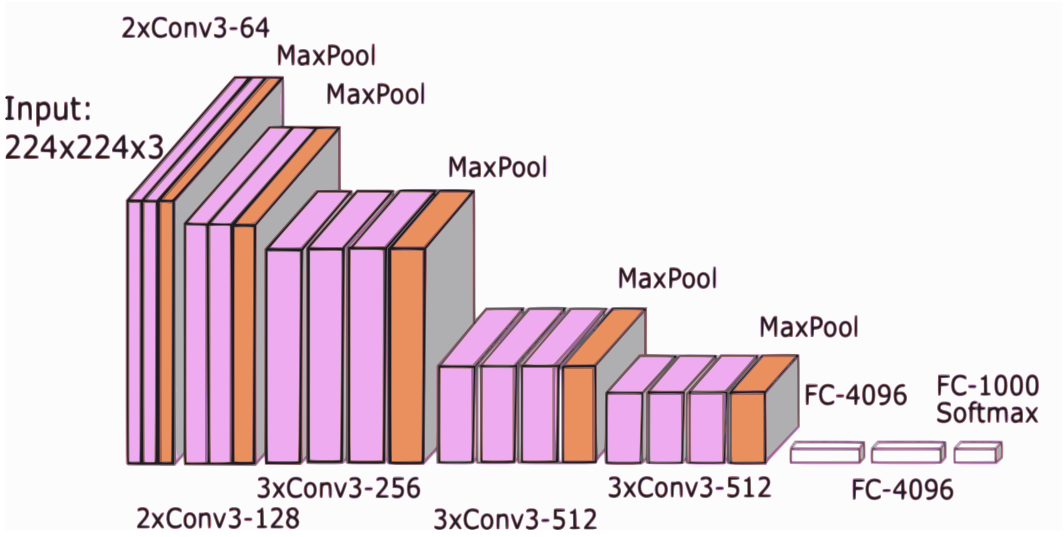


Figure 3.4 : The *VGGNet* model.

CNNs by randomly eliminating the units and their weights during training is also introduced in *AlexNet*. So long as [77] proposed a fascinating process called *ZFNet* in quest of visualizing what hidden layers see in a CNN, the prevailing techniques for understanding the behaviors of layer activations were poor. *ZFNet* is claimed to be an improved form of *AlexNet* and shows the performance results on different datasets. *ZFNet* architecture encompasses a deconvolutional layer that inversely maps the activities back to the input pixel space, thus feature activations can be traced with reciprocal inputs that help to have a better intuition to observe how features acts during training. A growing avalanche-like interest is still sustained in the vicinity of enhancing performance outputs of CNNs at the ILSVRC events. From this point of view, two object recognition winners of 2014 along with the numerous applicants supports this idea. One those claimers presents their study in [78] with the model called *VGGNet*. To reveal the power of depth for deep neural networks, 6 sequential models with an escalating quantity of hidden layers from 11 to 19 are compared, which use the same structure consisting of 3×3 convolutional filters followed by pooling operations of stacked layers as shown in Figure 3.4 for 16-layered model. *conv3-64* represents 64, 3×3 convolutional filters slide over the current layer and extract the new features. FC-4096 stands for the fully-connected layer containing 4096 units. Resembling to *AlexNet*, this model also utilizes $224 \times 224 \times 3$-pixel images as inputs. This study is the first to the example of employing 3×3 convolutional filters, and thus the network achieved the capability to copy the actions of larger receptive fields effect. 16-layered *VGGNet* contains $\sim$ 138 million connections between layers as

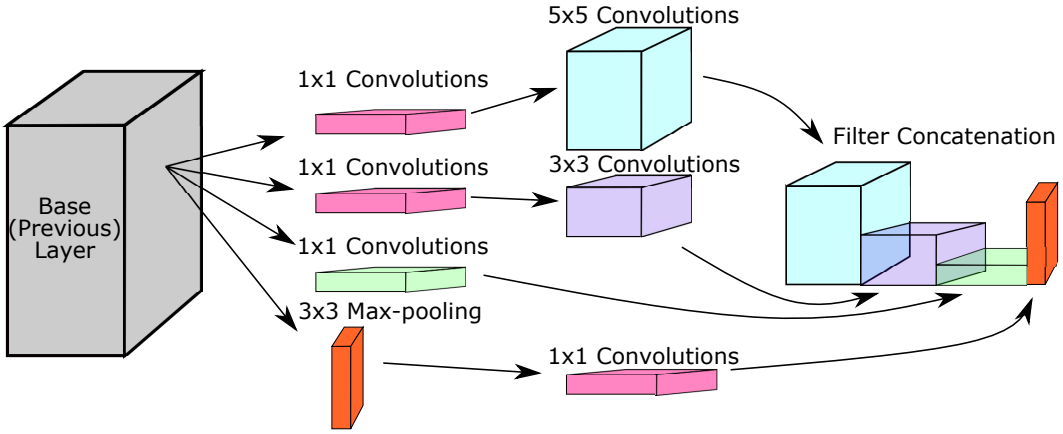


Figure 3.5 : The Inception module of GoogleNet with dimensionality reduction.

weights. In addition, the results of this study claim the impression that ensembles through networks involving more and more hidden layers yielding deeper networks improve the performance. The other one of the CNN architectures asserted as the winner of ILSVRC 2014 is explained in [79] and it is called *GoogleNet* with its peculiar building block structure called *Inception*. The *Inception* module with dimensionality reduction is shown in Figure 3.5 with its illustration for stronger mental grasp. If 1×1 convolutions in this illustration, which exist before 3×3 and 5×5 convolutions, are disposed of, then unsophisticated *Inception* module version is attained. This layer is also called bottleneck layer that reduces the number of features. *GoogleNet* has only 5 million parameters within its 22-layered architecture had by stacking *Inception* modules and it does not accomodate any FC layer in contrast to *AlexNet*-based architectures. This CNN model is trained by $299 \times 299 \times 3$ images. The objective of the *Inception* architecture is to achieve sparse structures from dense pieces of CNN features. This is accomplished by concatenating the independent convolutional and/or pooling schemes. The specific design of the *Inception* module enables *GoogleNet* to capture sparse patterns from feature maps through applying parallel filters with different receptive field sizes. However, this also raises the computational cost; bottleneck layer balances the cost by reducing spatial dimensions. The confusion about ILSVRC2014 winner occurred due to the crop sampling type for enriching the training images diversification. If models are trained by central-crop sampling, then the error scores of *VGGNet* and *Inception* are 8.70% and 10.07% respectively. On the contrary, training models with 10-crop sampling yields 9.33% and 9.15% errors respectively. Furthermore, [80] introduces the improved versions of *Inception*. *Inception-v2* is

modified by factorizing the classic larger convolutions into three 3×3 convolutions yielding expansion at each layer to increase feature diversity before next layer while diminishing the spatial resolution and *Inception-v3* adds regularization to training with batch-normalization (i.e., data-whitening) *Inception-v2*, which reduces the importance of auxiliary classifiers. In ILSVRC2014, a subtle and spectacular CNN architecture [81] called *ResNet* has beaten human-level performance on object recognition. With the goal of discovering an efficient way of training deeper neural networks with smaller error rates than the shallower ones, the *ResNet* block proposes to copy layers from the learned shallower model and to map the identity of these layers by skipping the intermediate connections via short-cuts likewise gated recurrent units. On account of this, the vanishing gradient and accuracy saturation (degradation) problems stemming from stacking layers on a plain neural network found answers, and it is alleged that using bottleneck convolutions same as *Inception* module is sufficient to access deeper efficient neural network architectures. The batch normalization operations are applied after each convolution and before activation. The experiments on ImageNet dataset are conducted with 5 models having various depth sizes starting from 18 layers to 152 layers, and the training image size is retained as the same with *Inception*. Excluding the classification layer *ResNet* does not have any FC layer and there are 60.2 million total parameters included in the 152-layered model.

There are also some types of distinct CNN architectures, which are derived from the given structures so far. For instance, *Xception* [82] is a modified *Inception-v3* architecture with 48-layered CNN consisting of convolutional and pooling layers with optional FC-layers. In conjunction with having 6 more layers than *Inception-v3*, it also converts its convolutions to the depth-wise separable convolutions by having the same number of parameters. In the study, it is shown that *Xception* takes over *Inception-v3* slightly in ImageNet dataset, but it is contended that in datasets with larger amount of images the difference becomes significant. The given CNN architectures so far are the current state-of-the-art and distinctive structures. In this thesis, we give performance results of *VGGNet*, *Inception-v3*, *ResNet*, and *Xception* models by fine-tuning them by adjusting some layers according to our dataset consistently. Another modified architecture called *Inception-v4*or *Inception-ResNet* [83] is obtained by combining the *Inception* module with *ResNet* simply by adding residual connections to the

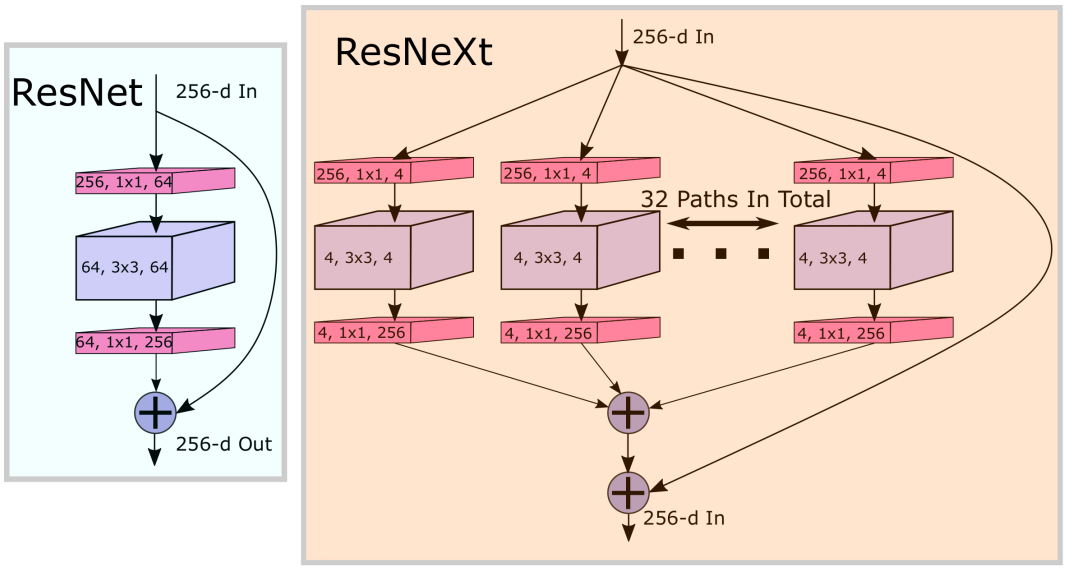


Figure 3.6 : The *ResNet* and its successor *ResNext* modules.

Inception-v3 modules, which yields better recognition performance with its more complex structure. In addition, an advanced version of *ResNet* module is introduced in [84] named as *ResNeXt*. A *cardinality* hyper-parameter is presented in the study as a set of individual, independent paths, which expands the block architecture that pursue the split-transform-merge procedure as *Inception* versions do. The study claims that the increase in cardinality gives better results rather than deeper or wider architectures. In Figure 3.6, *ResNeXt* block is given together with its primitive version *ResNet*. In spite of the fact that, there are definitions of the object detection as it is a composition of object recognition and localization, we just consider the object localization part of the object detection term that is to predict the location of the bounding-boxes in pixels surrounding the object within images, because we consider estimating the labels of images (object classification or categorization) and predicting the pixel coordinates spatially of the objects inside images as two separate problems. Similarly, [85] assigns discovering the spatial coordinates of the object within an imaginary rectangle surrounding called the bounding-box to the deep neural network based regression problem. The last layer of *AlexNet* is replaced with a regression layer and the network is trained to anticipate a ground-truth mask $m \in [0,1]^N$ for an image x as formulated in Equation (3.6):

$$\underset{\omega}{\text{minimize}} \sum_{(x,m)\in D} \| \, (Diag(m) + \lambda I)^{\frac{1}{2}} (DNN(x;\omega) - m) \, \|_2^2 \tag{3.6}$$

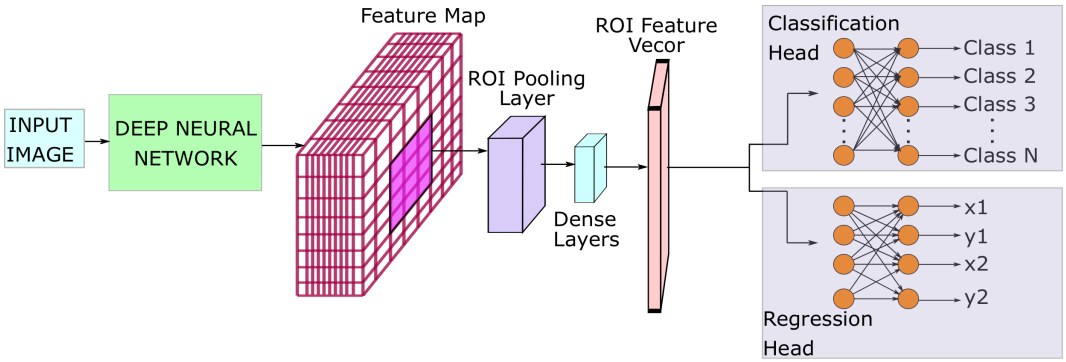


Figure 3.7 : Fast R-CNN operation scheme.

where $DNN(x;\omega)$ denotes the deep neural network output while x is given as the input image, and D represents the training set of images including the object bounding-boxes. From here, multiple masks are generated, then those are evaluated by non-maximum suppression to find out which one of these masks fit correspondingly to the ground-truth bounding-box. In [86], feature extraction is accomplished via an *AlexNet-based* CNN model called *R-CNN*, leading higher accuracy rates. To handle a small amount of annotated data issue, they use supervised pre-training on auxiliary datasets. The declared method has 3 steps; *i)* generating region proposals, *ii)* feature extraction for each region via CNN, *iii)* classification. The region proposals, 2000 of which are generated for each image by default, indicate a bounding box candidate with a possible object. After warping these region proposals into a fixed-length vector, then the *AlexNet-based* model with pre-trained parameters on ImageNet extracts features to be classified by support vector machines (SVM) per available class. As a consequence of this pipeline of initially training the CNN and then multiple SVMs for a huge amount of required disk space due to the 2000 region proposals per image makes *R-CNN* impractical. Our consideration about the separation of object detection by means of localization and object recognition subjects can be seen from Figure 3.7, which the workflow explained in [87] as being the improved version of *R-CNN* that is called *Fast R-CNN*. Same as the primitive version, this approach uses selective search [88] to produce tentative object locations, but instead of SVMs it only uses a single-stage CNNs whereas the most of the architecture is transferred, but mostly *VGGNet* is used this time. One single feature operation makes this model $9\times$ faster at training and $213\times$ faster at the test. The region of interest (ROI) pooling layer downsizes the proposed regions to a fixed size of 7×7, and then the network reshapes it to $F * 7 * 7$ vectors where F denotes the number of convolution filters. The network

outputs top-left $x-$ and $y-$coordinates, log height and width of the bounding box together with the class of the region. *Faster R-CNN* given in [89] goes one step further from [87] by substituting the selective search with a region proposal network (RPN). One can split *Faster R-CNN* into 3 consecutive steps as; *i)* feature extraction network usually comprised of convolutional layers that are initialized with pre-trained network parameters, *ii)* region proposal network (RPN) what uses the features and generated region proposal in the previous step, and *iii)* classification network usually composed of FC-layers that classify object proposals of each region and readjust the bounding box coordinates. The inserted RPN measures the "objectness" score by running a sliding window spatially with 9 anchors having 3 different aspect ratios and scales for each image over the feature maps extracted at the first step. Then, the values higher than a threshold are assumed to be probable bounding-boxes and these are evaluated by the classification network in the third step. If anchor box overlapping rates with ground-truth bounding-boxes have higher intersection over union (IoU) rate (simply $IoU = \frac{Areaof Overlap}{Areaof Union}$) than 0.7, then that anchor box is accepted to contain an object; otherwise, the with the highest IoU is used. Non-maximum suppression removes the unnecessary region proposals so that the final bounding-box is determined. In the complementary aspect, the study in [90] named as *YOLO*, alleges to use an *AlexNet-based* CNN model for both classification, and localization of object bounding-boxes. As can be seen from Figure 3.8 that input images with a resolution of 448×448 pixels are divided into 7×7 grids where each grid represents a prior box, and there are 2 bounding-boxes tested per image by default. The prior boxes only hold the information of the center location and regression head (linear layer) predicts the box size. The YOLO architecture output size is $S \times S \times (B \times 5 + C)$, where s shows the grip size, B denotes the number of tested bounding boxes, 5 is the additionally tested bounding-boxes for regression, and C represents the number of classes (by default number of image classes is 20). From here, one can calculate the number of different bounding-boxes evaluated for all grids as $7 \times 7 \times 2 = 98$. Together with the confidence ratio and non-maximum suppression operation, the irrelevant boxes are eliminated, and the softmax layers predict the classes. In addition to an improved version called *YOLOv2* based on this architecture, which mostly attempts to overcome the issues of detecting small objects and unusual aspect ratios. Furthermore, Single shot multibox detector (*SSD*) [91] declares a novel bounding-box detector, which does not require an

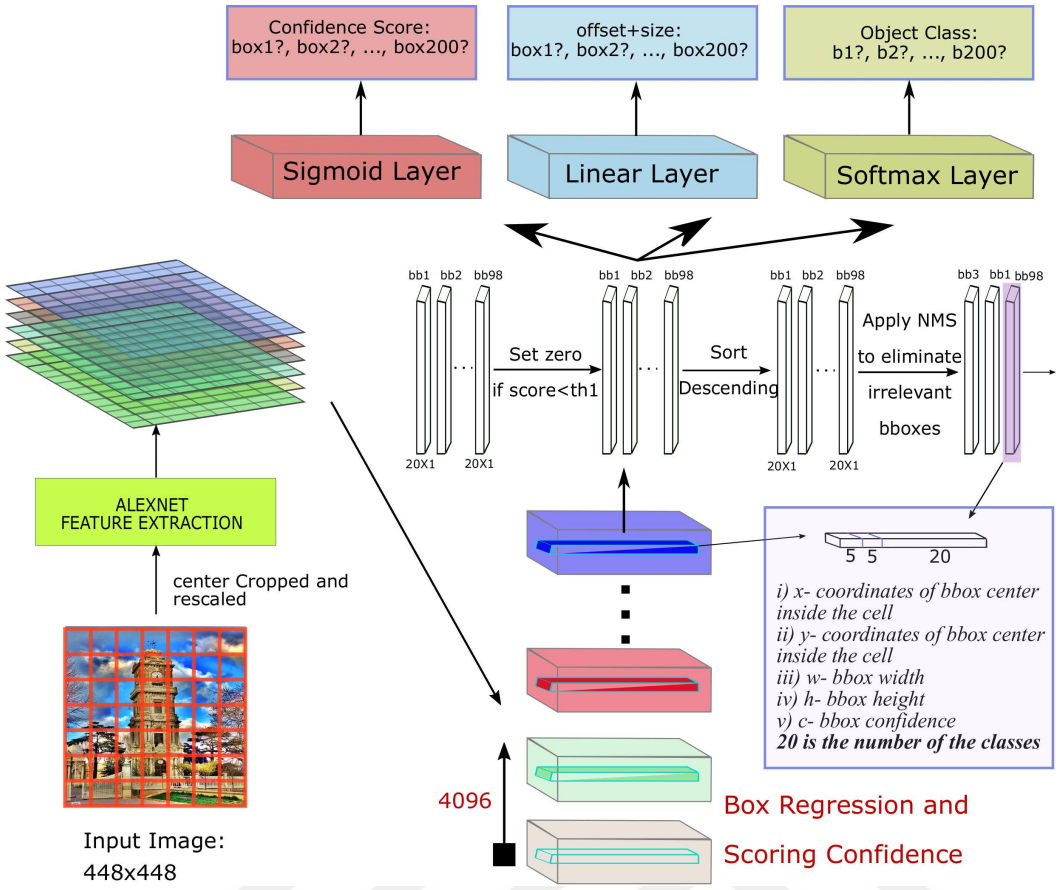


Figure 3.8 : YOLO object detection procedure.

RPN and ROI-pooling makes it very fast almost 60fps. *VGGNet* is modified by adding 6 extra convolutional layers to be used as the base architecture. The input images are rescaled to a dimension of 300×300. The output of the base layer and extra feature layers are concatenated at the FC-layer at the end of the network. This allows detecting the objects at different scales. The loss function is composed of a confidence loss for classification and location loss for regression that is given in Equation (3.7) as follows:

$$L(x,c,l,g) = \frac{1}{N}(L_{conf}(x,c) + \alpha L_{loc}(x,l,g)) \tag{3.7}$$

where x is the matching score, N denotes the number of matched default boxes, c is the class confidence scores, α shows the weight term, g represents the ground-truth parameters, and l is the predicted box. The architecture given in this study is illustrated in Figure 3.9. In short, SSD process can be summarized in 4 steps as; *i)* feed the image forward through the convolutional layers producing several feature maps at different scale in an ascending order, *ii)* by employing a 3×3 convolutional kernel a number of default bounding-boxes (same as the *Faster R-CNN* anchors) are evaluated in each feature map per location, *iii)* the bounding-boc offset values together with the

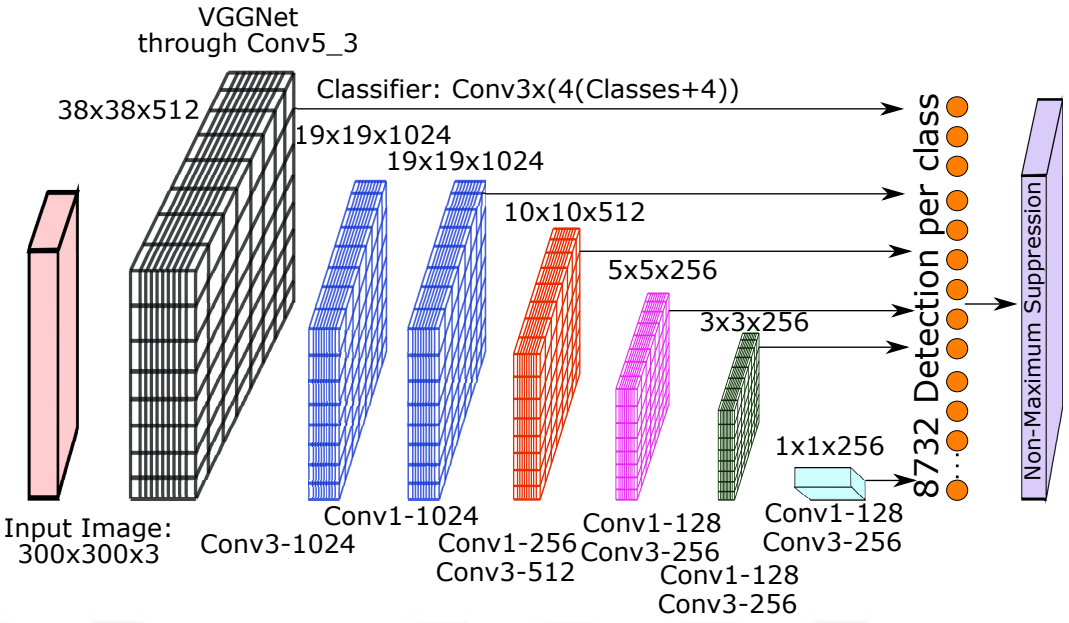


Figure 3.9 : SSD object detection procedure.

object classes are predicted for each boxes, and *iv)* compare the predicted boxes with ground-truth using IoU while training and the highest score is specified as including object piece inside. To remove overlapping and redundant boxes non-maximum suppression is applied. In advance of detecting objects, feature pyramid networks (*FPNs*) [92] highlight the employment of CNNs like feature pyramids including low-level feature maps that collaborate to detect small objects. The network uses the top-down pathway; thus semantically powerful feature maps are accumulated in two branches, and this ensures the scale invariance. The first branch is composed of convolutional and pooling layers as usual. And the second branch, which takes the outputs of the first branch as inputs, applies nearest-neighbor upsampling by its $256 - channels$ to increase the resolution again to the original image. The connection between two branches is established via 1×1 convolutions before an addition like a residual connection. The study declared in [93] called *RetinaNet* is a single stage method including two task-dependent sub-networks towards object detection and classification. The *RetinaNet* architecture consists of a feedforward *ResNet* model followed by an *FPN*. At the end of this model, the sub-networks are attached; one for anchor box classification and the other one for regressing from anchor boxes to ground-truth as demonstrated in Figure 3.10. The imbalance between background and foreground classes is argued to be the reason for the poor performance of single stage methods for object detection. In consequence, they contend a Focal Loss term, which

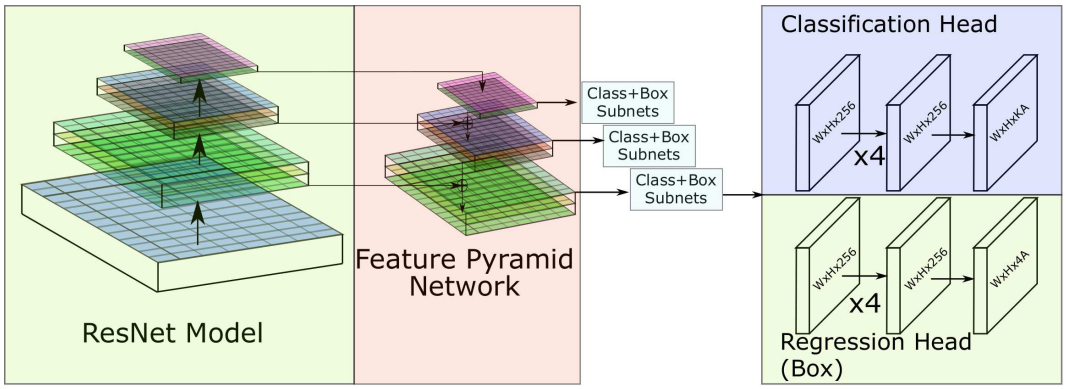


Figure 3.10 : The RetinaNet architecture flow-chart.

includes a factor $(1-p_t)^{\gamma}$ to the regular cross entropy loss ($CE(p_t) = -log(p_t)$) as given in Equation (3.8):

$$FL(p_t) = -\alpha_t(1-p_t)^{\gamma}log(p_t) \tag{3.8}$$

where $p \in [0,1]$ is the estimated probability of the model, $\alpha \in [0,1]$ is a weighting factor, $(1-p_t)^{\gamma}$ demonstrates the proposed modulating factor with tunable focusing parameter $\gamma \geqslant 0$. The benefits of this focal loss are further displayed. In this thesis, we chose *RetinaNet* to detect objects and the details about implementation are provided under the "Results" section of this chapter.

There also exist remarkable models and CNN structures in literature, so that for further studies we refer the reader to [94], [95] (this study can be assumed to be the ancestor of *Inception* modules), [96] (a rare kind of model working to serve for 3 tasks; detection, localization, and recognition of objects), [97], [98], [99], [100], [101], [102], [103](predicts masks for detected objects, mostly used for segmentation), and [104]. However, we confined this section with the explained studies as long as this thesis includes the object detection and recognition subjects. As per what we have seen so far, those unique building blocks and architectures enlighten the path during the journey of the deep learning researchers towards achieving robust, reliable, safe, and even higher performance than human-level from machines and robotic systems.

3.3 Datasets

Comparable to humans, vision datasets imply the most prominent processing capabilities like data retrieved by eyes in order to extract valuable information. Because the primary focus of this thesis is to manipulate visually recognized objects to

accomplish a sensible motion, we will give insight about image datasets (and/or word datasets that are linked to the image datasets) in this section, which is conceded as benchmarking datasets.

Image datasets can be considered in two groups; labeled and unlabeled/raw, which are relevant for supervised learning (classification) and unsupervised learning (clustering) tasks, respectively. Furthermore, semi-supervised and reinforcement learning algorithms can be applied to both types of datasets. Additionally, much more effort is required to attain labeled image datasets than unlabeled ones. Moreover, the quantity and quality of data affect the performance of machine learning systems directly. It is worth to note that the importance of datasets for learning algorithms are commensurate with human sensory organs.

As mentioned earlier in this study, ImageNet is a huge dataset consisting of 15 million annotated images in 22000 categories, and ILSVRC is an annual competition held under the subjects of object localization, object detection, object detection from the video, scene classification, and scene segmentation. In the challenge, 1.2 million images in 1000 categories are employed to measure the performance of the submitted algorithms. In the ImageNet dataset, images are labeled according to the WordNet hierarchy [105]. Likewise, a competition is run using Microsoft Common Objects in Context (in short COCO) dataset [106] every year. COCO provides a total of 330000 images (more than 200000 images are labeled) in 80 object and 91 stuff categories in addition to 250000 people with keypoints. There are 1.5 million object instances and 5 captions for each image. In COCO, the competitions are executed in the subjects of object detection, instance segmentation, image captioning, and person keypoints localization. An older example of image dataset challenges, the PASCAL Visual Object Classes (VOC) [107] was held from 2005 to 2012 as a yearly competition. PASCAL VOC is a benchmarking dataset for assessing the performance of object category recognition. The initial dataset was composed of 1578 labeled images in 4 classes as long as there was 11530 region of interest (ROI) annotated images in 20 categories in 2012 in this dataset. Caltech-101 [108] and Caltech-256 [109] consist of 9146 and 30607 images in 101 and 256 classes, respectively and each class includes various numbers of labeled images ranging from 40 to 800 with the dimension of 300×200 in pixels. CIFAR [110] is derived from 80 million tiny images dataset [111]

by labeling 60000 for 10 classes called CIFAR-10, and another 60.000 for 100 classes, which are composed of 5 classes under 20 superclasses called CIFAR-100. One of the biggest publicly available image dataset [111] contains approximately 80 million colored images with the dimension of 32×32 pixels with weak labels which are listed within WordNet hierarchy. Yale-CMU-Berkeley (YCB) [112] dataset presents 77 classes of objects relevant to robotic manipulation research. YCB contains 600 high-resolution colored images, 600 colored depth images and five sets of textured 3 dimensional geometric models with mass values of objects per category.

n addition to ADORESet containing both real and synthetic images, we also give knowledge about significant synthetic image datasets to reveal the potential of our dataset. [113] provides a synthetic image generator and proposes a pipeline to bring about better results than real-world data when working only with synthetic images. However, this study compares the effects solely for vehicle detection tasks. Similarly, the synthetic collection of imagery and annotations (*SYNTHIA*) incorporates only synthetic outdoor images, which are captured from a virtual world with pixel-level labels. There are over 200000 high-definition images grabbed from videos and over 20000 high-definition separate images in the direction of semantic segmentation and scene understanding from driving simulations. The results in *SYNTHIA* indicates that hybrid dataset approach also contributes to semantic segmentation of objects. Furthermore, SceneNet [114] dataset provides labeled synthetic 3D indoor scenes for 5 categories; bedroom, office, kitchen, living-room, and bathroom consisting of objects from 50 to 150.

It is noteworthy that there exist image datasets in the literature except for the given datasets above, which are concentrated on particular subjects such as annotated hand images [**?**], street view house number images [115], brand logos [116], handwritten characters [74], faces [117] and so on. Additionally, Places [118], Places2 [119], and LabelMe [120] datasets are constructed using outdoor images which contain labeled and/or weakly labeled images. Concisely, it is conceptual for deep learning research to focus on a compact problem, collect raw data and clean it, then ameliorate existing algorithms by fine-tuning or deriving depending on data to acquire plausible

results. Reasonably, we propose *ADORESet* that provides opportunities of transition and flexibility for real world and simulation environment applications.

3.3.1 A hybrid image dataset towards bridging the gap between real and simulation environments for robotics: ADORESet

Robotics research problems involving machine vision are generally carried out using real and simulation images, separately. These images routinely belong to the categories of objects namely tableware, glassware and similar kinds of objects, which may remain on desktops and tables, in the form of relatively small, graspable, pushable states. Due to the fact that the essential objective of robotics research with regard to achieving results at human-level or better for vision and control tasks, using application-specific algorithms is convenient. Thus in this study, we propose annotated desktop objects real and synthetic images dataset and name it as *ADORESet*, which contains data from both real-world and simulation environment. From this perspective, the main reasons behind proposing ADORESet, as a hybrid image data set including both real and synthetic images, are suggested as follows; able to eliminate incompatibilities between real and simulation environments by training once and running the model weights everywhere, ready to minimize experimentation time wasted during adjusting the simulation model to the real world, consisting of relevant object categories for dexterous manipulation, grasping, detection and recognition. Precisely, the diversity of the dataset samples is another crucial factor, which directly influences performance, such that raising the amount and diversity of training data by data augmentation has been a prerequisite for deep CNN models. Figure 3.11 illustrates the comparisons of the explained image datasets including *ADORESet* relatively in the logarithmically scaled 3D space, where the vertical axis shows the total number of images, the horizontal axes indicate the average number of images per class and the total number of categories, respectively. Even if the primary goal of machine learning algorithms is to obtain knowledge from data using algorithms, the quality of data regarding quantity, labels, missing samples, variations, noise, outliers, invalid instances, etc. has a significant influence on resulting models. Therefore, densely annotated ADORESet provides a competent number of images per class for machine vision based problems in robotics such as object detection and recognition, object tracking and manipulation.

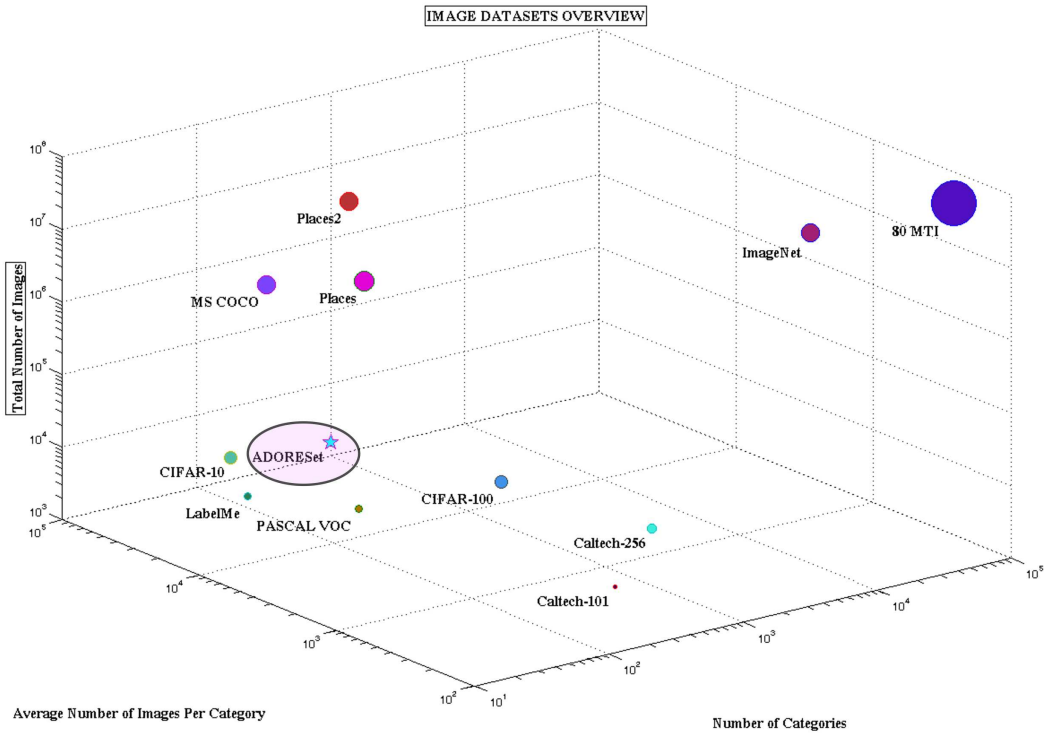


Figure 3.11 : Datasets in logarithmic scale according to total number of images, average number of images for each category and total number of categories.

Such a dataset consisting of real and synthetic images maintains flexibility concerning developing models both for real-world and simulations, then enables deployment of these models for experimentation directly. ADORESet should be of interest to the field of robotics researchers by means of its hybrid form, compactness in terms of lightweight and relevancy to robotics applications such as detection and recognition, grasping and dexterous manipulation of objects.

In order to construct ADORESet, we start by downloading instances via image search engines. Afterwards, an adequate number of images of relevant classes are generated within the simulation environment. The annotated and resized data received from both sources are processed using ITUrk graphical user interface (GUI). Successor objects are also labeled to build statistical information about the relationships of the objects within the dataset. For example, the connection between monitor, keyboard, and mouse can be directly achieved using this practical information. Figure 3.12 shows the pipeline of these steps.

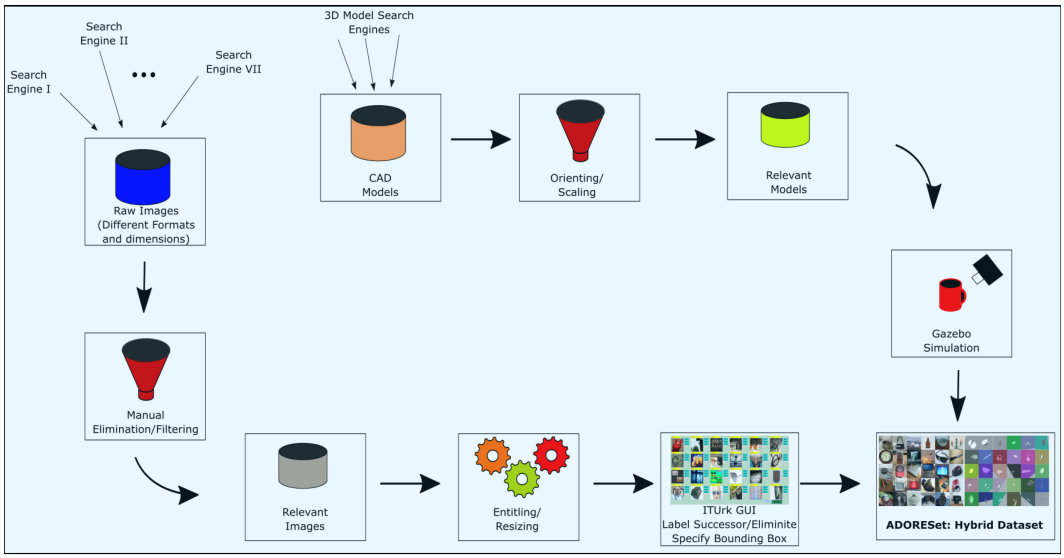


Figure 3.12 : The operations during the construction of the *ADORESet*.

3.3.1.1 Collecting images from wild web and preprocessing

The object categories in ADORESet, which are provided in Table 3.1, are specified considering the robotics applications. It is unquestionable that these objects have been part of everyday life in the last 3 decades. With the ambition of building this dataset using the wild web, we endeavored about 390 query terms or word pairs via 7 image search engines. Principally, the multi-language wild web search is performed according to the brand, gender, model, type, color, age, season, material, state, and relation. Subsequently, inappropriate raw images are eliminated manually with regard to criteria such as the light effects and conditions, noise, distance and angle, visibility which influence the quality of the dataset. Then the remaining images are entitled to the following rule: The first three numbers show the category starting with 0, and the last five digits indicate the index number of image in that category starting with 0, e.g., "01700754 is the 754^{th} image of the *laptop* class". Concurrently, all images are resized to the same dimensions. Eventually, ADORESet is a new richly-labeled dataset consisting of 75000 colored real images with the size of $300x300$ pixels for 30 classes including the bounding-box coordinates of all objects. The real images are stored in JPEG compression format, which takes up approximately 1.3 gigabytes space in the hard-drive.

Table 3.1 : ADORESet Object Classes.

1 Ashtray	2 Bag	3 Book	4 Bottle	5 Bowl
6 Can	7 Candlestick	8 Clock	9 CookingPot	10 Cup
11 DeskLamp	12 Eyeglass	13 ForkSpoonKnife	14 FryingPan	15 HeadWear
16 Keyboard	17 Laptop	18 Monitor	19 Mouse	20 Pen(cil)
21 PhotoFrame	22 Shoe	23 SmartPhone	24 Speaker	25 Teapot
26 Telephone	27 Vase	28 Wallet	29 WebCam	30 WristWatch

3.3.1.2 Image generation from simulation world

Similar to gathering real images, the process of image generation from simulation world starts with downloading computer-aided design (CAD) models of the objects from the wild web. For each object class, five different CAD models are downloaded, and their file formats are converted to STL which is appropriate for universal robot description files (URDF). Since they are acquired from distinct sources, their orientation, scale, and origins are not correctly defined. Initially, every model has oriented in a way that normal vector of the meaningful side of the object is parallel with the z-axis. Next, the objects are scaled to their real-world dimensions. Lastly, the origins are relocated to bottom centers of the CAD model. Textures are not attached to the models, and the colors are allowed to change with the glow of the simulation world sun. Consequently, ADORESet includes 750 synthetically generated images per category carrying the same properties as real images.

There are two critical variables in the simulation world which affects variations and the quality of the images, light color and 6D pose of the camera. In GSE, the light is adjusted with the sun model and 30 images are captured for any sub-object couple in the same conditions. After completing image acquisition, old sun model is deleted, and a new sun with random color values is spawned. 6D camera pose consists of 3 position and 3 orientation variables. It is assumed that two virtual half spheres are created around the object with the radii of r and R, and the camera is located on their surfaces. So the distance between the camera and the object is similar for each object class depending on its average dimensions. For instance, the minimum distance (r) between the camera and the object are set to 0.2 m for wristwatches while it is 0.4 m for bowls. The environment with the half sphere is drawn schematically in Figure 3.13. To calculate a random point on the half sphere surface, an arbitrary unit vector **s**

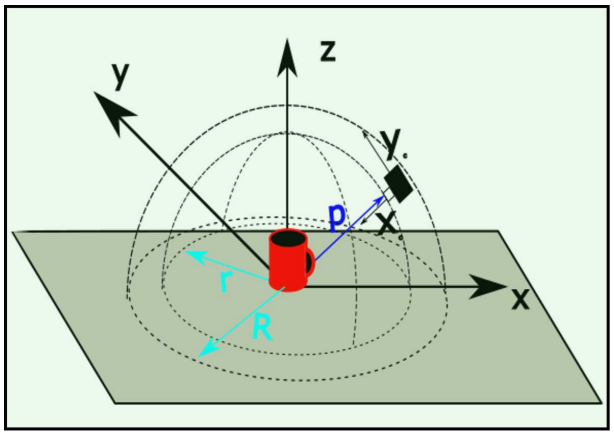


Figure 3.13 : Schematic view of simulation environment with frames and variable definitions.

is defined as given in Equation (3.9):

$$\mathbf{s} = [rand(-1,1), rand(-1,1), rand(0,1)]^T \tag{3.9}$$

, where *rand* denotes the random function between given argument values. It is worth to note that **z** vector is restricted to positive numbers which restrain the position of the camera on the upper half of the sphere. The position vector of the camera **p** can now be easily calculated by using known values of r and **s** as in Equation (3.10):

$$\mathbf{p} = (c_1 rand(-1,1) + r) \cdot \mathbf{s} \tag{3.10}$$

The constant c_1 stands for the maximum distance between the object and the camera R. The opposite direction of the position vector defines the pointing direction of the camera orientation $\mathbf{x_c}$. To use the vector in frame definition, normalization is applied as in Equation (3.11):

$$\mathbf{x_c} = -\frac{\mathbf{p}}{\|\mathbf{p}\|} \tag{3.11}$$

Because the calculated $\mathbf{x_c}$ vector guarantees that the object is on the image plane, other orientation vectors can be selected as arbitrary vectors meeting the orthonormal condition. So $\mathbf{y_c}$ is calculated ensuring the dot product with $\mathbf{x_c}$ results in zero as in the following equation.

$$\mathbf{y_c} = [c_2 x_{cy} + c_3 x_{cz}, -c_2 x_{cx}, -c_3 x_{cx}]^T \tag{3.12}$$

Three components of the $\mathbf{x_c}$ vector are denoted by x_{cx}, x_{cy} and x_{cz}. The last vector to form the orientation or rotation matrix is $\mathbf{z_c}$. It has to be a perpendicular vector to other two and is calculated as given in Equation (3.13).

$$\mathbf{z_c} = \mathbf{x_c} \times \mathbf{y_c} \tag{3.13}$$

Figure 3.14 : Example images for all object categories generated in GSE.

Random sun spawning and 6D pose generation are implemented in a ROS node. Every image is grabbed from a specific 6D pose. The light source is changed for every 30 images because of the speed of deleting and spawning the light. 5 different CAD models are utilized for each object which produces a total of 750 images for each object class. Example images per class are shown in Figure 3.14.

3.3.1.3 ITUrk GUI

Although the wild web supplies an excessive amount of data, it may cause problems when it is used with deep learning algorithms directly due to the lack of quality. In fact, many images tagged with inconsistent keywords or indistinguishably tiny sized objects remain within the images. To overcome these complications, in most cases, crowdsourcing tools are employed to label the data. There are such tools that are designed for a more general social experimental task which are also known for labeling data called Amazon Mechanical Turk (AMT) [121]. Furthermore, the software as reported earlier can be designed to collect a broader range of information than the one obtained by only labeling. So that the gathered data can be extended to have a knowledge of the labeled object location in the image plane. It also allows to label the

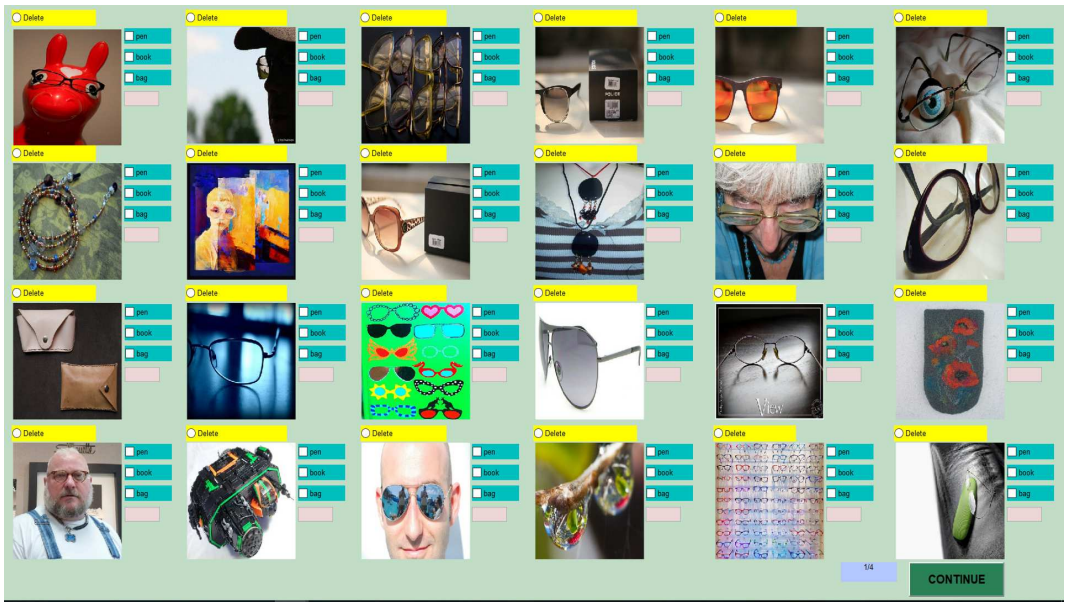


Figure 3.15 : ITUrk GUI with the images from eyeglass category.

successor objects remaining in the scene. In this work, a simple GUI is designed and implemented to annotate the data, the bounding box location and the successor object.

The GUI is designed to process 24 images on a single page to increase the speed while keeping them visible enough for the user. Each object class is loaded to the GUI first. Then the user is suggested to delete irrelevant images about the object class by selecting them from the delete buttons over the images. At the same time, the user clicks on the complementary object name if a successor object exists. Three most expected successor names are readily given as click buttons. However, the user can add more associated items by writing the name of it to the text-box under the successor names. After completing elimination and labeling successor objects, the continue button starts the bounding box selection. The user selects left-top uppermost bounding point with mouse left click. Similarly, right-bottom lowermost bounding points are chosen by the right mouse click, and it accomplishes the bounding box selection for the current active image. The active images are marked with red delete buttons. After the bounding box selection of it is completed, the next undeleted image becomes active. Finally, completing bounding box selection starts a new page with new 24 images. The GUI is implemented in MATLAB. The screenshot of the GUI is given in Figure 3.15. A total of 75000 real images belonging to 30 object classes are filtered through ITUrk as available images for deep learning algorithms. These colored images are resized to a dimension of $300 \times 300 \times 3$ pixels which is the same

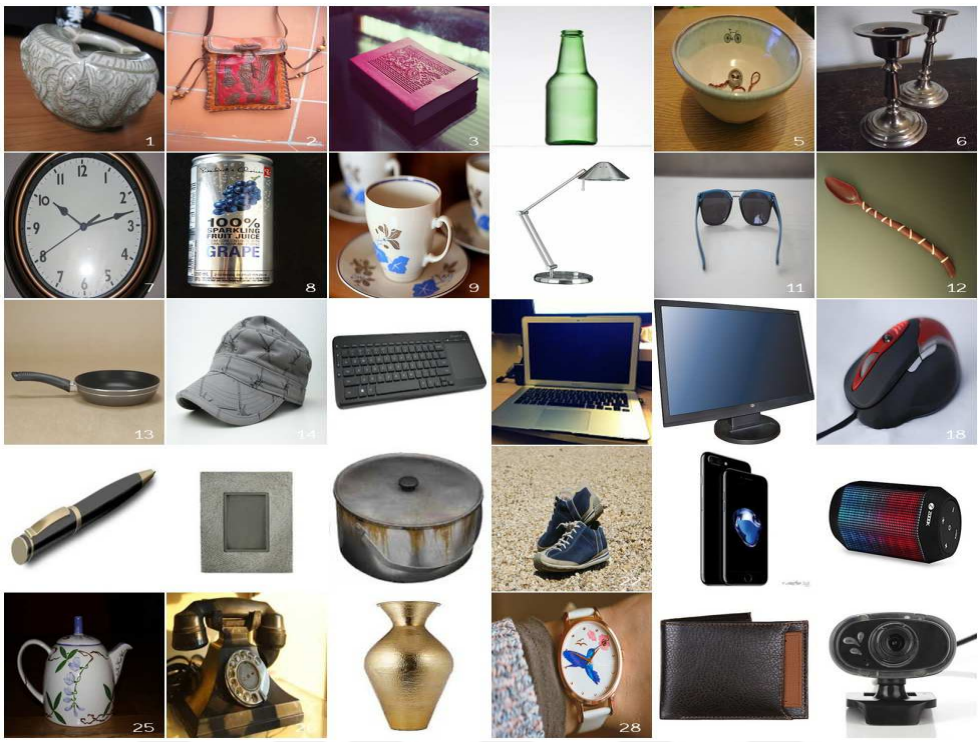


Figure 3.16 : Resized and labeled wild web images with instances from all categories.

as the images from simulation world. The user can process 24 images in one single page within 2 minutes approximately. The initial 40 seconds are spent in annotating and successor labeling part and the remaining time is devoted to the bounding box selection. Moreover, the perspectives and cylindrical objects may reduce the speed of the process and cause the failure of the human bounding box specifiers. The example images from each of the object classes are shown in Figure 3.15.

3.3.1.4 Distinctive properties of ADORESet

The underlying philosophy beneath the machine learning systems is to have a dataset which has as many varieties as possible and then to develop intuition using supervised, unsupervised or reinforcement learning algorithms from data. Notably, researches on robotic arms for non-industrial daily use and humanoid robots are increasing in the last years. Both real and simulation world applications give successful results in areas such as perception, recognition, gripping, grasping, moving, and manipulating, separately. To make systems intelligent, the training data has to be compatible with the environments where the test sessions will be conducted. Accordingly, concerning the compensation of these requirements, ADORESet is composed of hybrid images

for 30 object classes, which may exist mostly on desktops. Following the labeling and elimination operations, some images are exposed to distortions because of resizing, that yielded variations for the dataset images, which is one of the most desired properties for an image dataset. Since, there is adequate number of images per category, each class of ADORESet is also convenient for sub-category classification. Unlike datasets as mentioned earlier, which consist of single and centered objects per image, ADORESet contains complex images including multiple objects, which makes it a more challenging dataset, besides comprising different forms of objects that transformed in decades. Our dataset also includes a satisfactory number of centered and salient images that can be readily separable from the background. Another property that discriminates ADORESet from the other datasets is to provide useful information about the relationships between objects by extracting knowledge from the statistics of the labels, which is obtained by annotating the successor objects in the dataset.

3.3.1.5 Statistical analysis of ADORESet and semantic relation between objects

The object classes, which are included in the ADORESet, are chosen from commonly used items in everyday life and mostly located around or on desktops. In addition to this, the objects are related to each other depending on their usage area, appearance similarity, and ideal locations. Some of them are used in similar or completely same tasks. For instance, *telephone-smart phone* are used for communication purposes and *pot-pan* are used for cooking. Additionally, some tasks have multiple objects which completes each other, such as *mouse-keyboard*, *cup-teapot*, *cup-bottle*. Besides, physical appearance is another important issue, and for some object class couples, it is not occasionally distinguishable as in the case of *bowl-vase* and *pan-pot*. Furthermore, specific items are usually placed close together. For example, it is strongly probable that a *fork* may be encountered near to a *bowl* or a *cup*. It is worth to consider that the object classes consist of not only one object but also multiple very similar objects. The cutlery items object class, *Fork/Spoon/Knife*, aggregates 3 eating utensils as an example. Successor objects are detected in randomly collected images from the wild web to identify semantic relationships between them. It may bring proper information to researchers from robotics area, particularly in manipulation planning. To provide the information, existence frequencies of successors for each object classes are illustrated

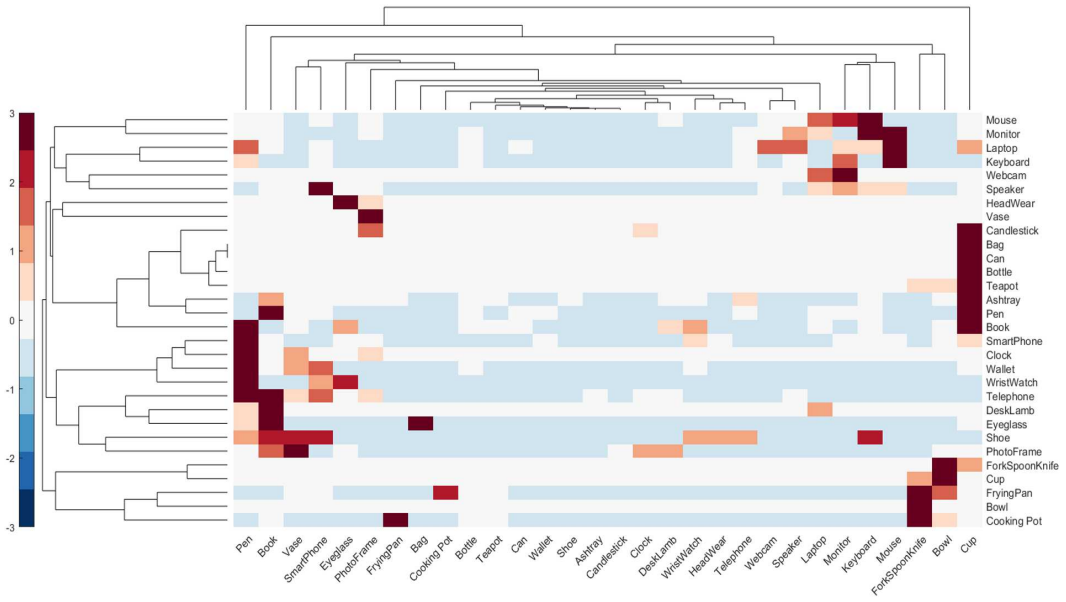


Figure 3.17 : Relations between object categories. (Darker color means more relationship between objects.)

as a color matrix in Figure 3.17. The main object classes are indicated in row entities, and their successor images are given in columns in Figure 3.17. Since the object is not a successor for itself, the frequency is assumed to be zero. The columns and rows are arranged in an order that strictly related objects are closely aligned. All values are standardized along the rows to emphasize the relations. As a result of standardization, relation scores of the objects are colored according to colorbar given on the left side of Figure 3.17. Thus, for example, *bowl* is the most frequent object in *cup* images. On the other hand, it is worth to consider that graph is not symmetrical. As a result of this, *cup* is not the most existent object for *bowl*. So a robot can interpret that if a *cup* is in the scene probably a *bowl* can be seen, however, if a *bowl* is seen it cannot be said that a *cup* is in the area.

The statistical analysis shows the relation between the object classes numerically. Robots empowered with vision make use of this meaningful information to increase inherent capabilities of object search or the accuracy of object detection under the influence of weak lighting or occlusion. Vision algorithms may estimate where to look for a particular object in a massive operation space. An occluded object can be identified more precisely with the assist of detected successor objects. The analysis helps manipulation planning tasks through clustering similar objects as well. Statistical results guide to the robot to place complementary items together in a meaningful way.

As a consequence of having richly annotated 3250 images per category and containing an equal number of real (2500) and synthetic (750) images individually for each class put *ADORESet* one step ahead amongst others.

3.4 Convolutional Neural Networks

The necessary amount of information about deep neural networks is given so far in the context of object detection and recognition tasks. It is clear that the recent approach employs convolutional neural networks by a majority and the building-block of ANNs, backpropagation procedure, and optimization steps are clarified above. Hereafter, we will explain the remaining components of the CNNs with a focus of processing the image data, which are activation functions, convolutional and pooling layers together with the hyper-parameters required to be optimized and the batch normalization will also be expressed in this section.

3.4.1 Components of convolutional neural networks

Since the CNN models can simply be formed by stacking layers in a sequence, every layer in these models transmits signals to another via a differentiable function called activation function. Activation functions maintain the non-linear mappings between the layers and so that it gives the ability to approximate non-linearities to the CNN model. Therefore, primitive activation functions such as binary step function $f(x) = 1$, if $x > 0$ and linear activation $f(x) = constant * x$ do not produce satisfactory results because of their gradients will yield to $f'(x) = 0$ and $f'(x) = constant$, respectively. In consequence, during backpropagation, the gradients of these functions do not improve the error rates.

As can be defined the mission of activation functions, they adjust the magnitude of the weighted inputs and provide nonlinearity to the deep neural networks while they transmit signals. For this reason, an activation function has to be in a non-linear form to be able to generate non-linear mappings between layers and must be differentiable continuously. As long as neural networks are universal function approximators, deep neural networks try to make sense from complicated problems in higher-dimensional datasets. In Figure 3.18, the frequently used activation functions and their gradients are displayed.

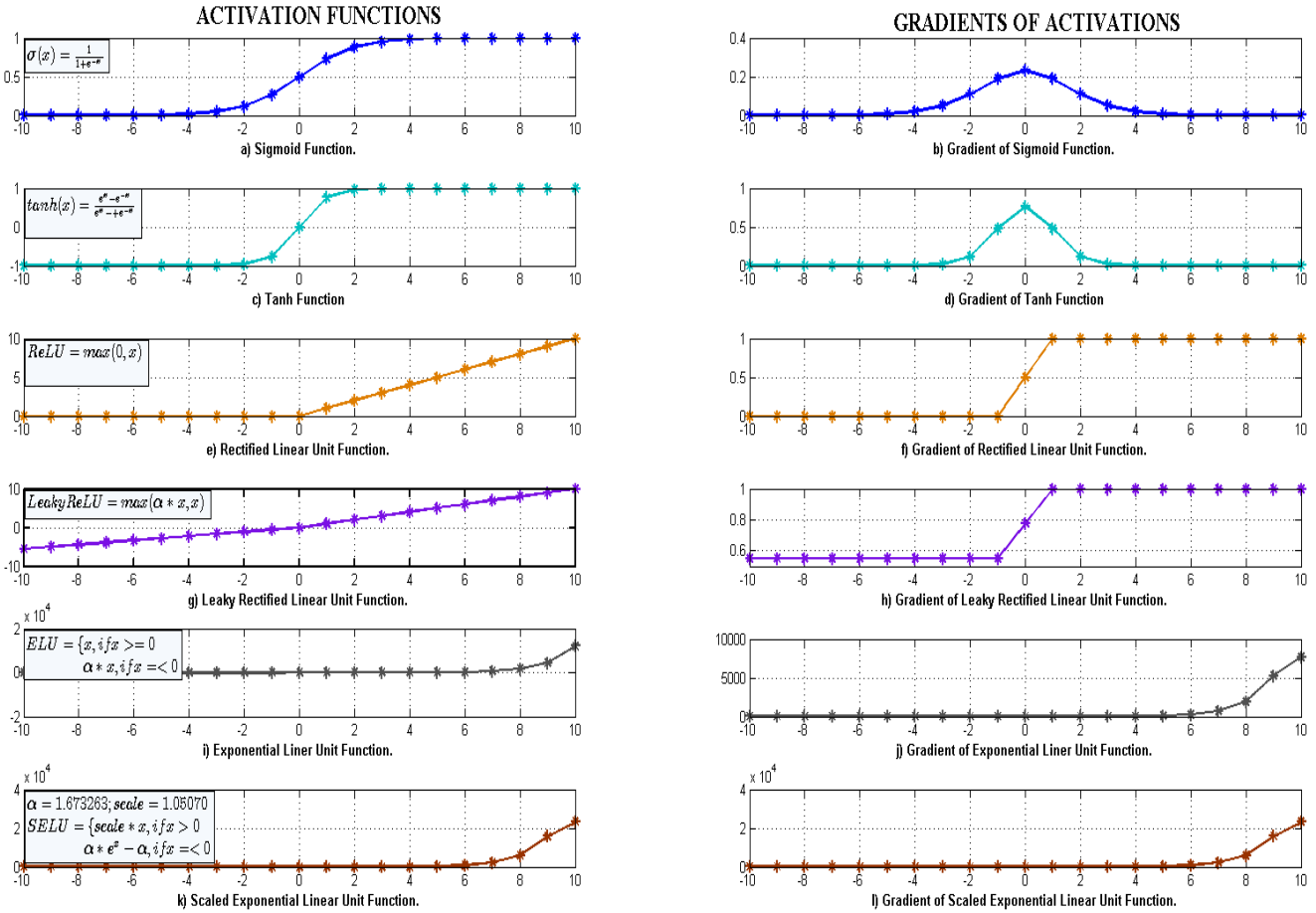


Figure 3.18 : Commonly used activation functions and their gradient correspondents for $x = [-10, 10]$: a) Sigmoid, b) Gradient of Sigmoid, c) Tanh , d) Gradient of Tanh, e) ReLU, f) Gradient of ReLU, g) LeakyReLU, h) Gradient of LeakyReLU, i) ELU, j) Gradient of ELU, k) SELU, l) Gradient of SELU.

Sigmoid or logistic function $\sigma(x) = \frac{1}{1+e^{-x}}$ is a smooth S shaped function within a range between $[0,1]$ whilst the gradient simply allows the range between $[-4,4]$. This means that the values outside the $[0,1]$ range do not influence the outputs as much and the values inside this range give rise to large changes in the outputs. One can see that from the gradient of the sigmoid function, during backpropagation, the values inside the $[-4,4]$ range become very small and the contribution to the learning of the network is restricted as well. The sigmoid is able to maintain non-linearity, and it is differentiable continuously; however, the convergence process is very slow besides sigmoids to saturate and kill the gradients, namely, it causes vanishing gradients problem. The tanh or hyperbolic tangent function $tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$is a rescaled version of sigmoid function. Unlike sigmoid function, the tanh function outputs are zero-centered since it is symmetric over the origin, ranging between $[-1,1]$. However, the remaining issues are the same as sigmoid function; the saturation and vanishing gradients. In deep learning applications, the rectified linear unit (ReLU) [122] $ReLU(x) = max(0,x)$, which is a non-linear function by means of having ability to backpropagate the errors, is vastly used because it plainly avoids and solves the vanishing gradient problem. But the mean of the ReLU function is not zero-centered. The negative inputs are not triggered in this function, so only the positive-valued neurons are activated that result in a sparser network than the original. However, this effect also appears in the gradients of the ReLU function that gives all zero when the received signal is negative so that this is a potential to dead neuron, which is never activated. To overcome this problem an improved function LeakyReLU [123] $LeakyReLU = max(\alpha * x, x)$ (where α is the coefficient allowing small gradient when it is negative) is declared, which presents a small sloped line in the negative side to keep the updates alive. Exponential linear unit function (ELU) [124] given in Equation (3.14) is another function that takes care of vanishing gradient problem and its mean is very close to zero that facilitates the network and reduces the computational cost as well, which result in the learning speed up.

$$ELU(x) = \begin{cases} x, & \text{if } x >= 0 \\ \alpha * x, & \text{otherwise} \end{cases} \tag{3.14}$$

By the reason that ELU is an exponential function, it does not saturate. The negative part of the ELU function behaves as bias term. Scaled exponential linear unit function (SELU) [125] is an upgraded version of ELU with two fixed parameters as given in

Equation (3.15).

$$SELU(x) = \lambda * \begin{cases} scale * x, & \text{if } x > 0 \\ \alpha * e^x - \alpha, & \text{otherwise} \end{cases} \tag{3.15}$$

These parameters $\alpha = 1.6732$ and $scale = 1.0507$ are constant so that they cannot be backpropagated, which mean they are not the hyper-parameters to be optimized. As can be seen, its functional drawing that this function diminishes the variance for negative inputs and raises the variance for positive inputs. The decline and gain effects are sharper for extremely negative and near-zero input values, respectively. In addition, the projections of fluctuation are surrounded from above and below, so the gradients cannot explode or vanish. As companion to *dropout* method, [126] proposes a feed-forward technique called *maxout*, which contemplates to simplify the optimization by *dropout* and the accuracy of its approximate model averaging technique. As a new kind of activation function, for the input $x \in \mathbb{R}^d$ the *maxout* runs the following function as given in Equation (3.16):

$$h_i(x) = \max_{j \in [1,k]} z_{ij} \tag{3.16}$$

where $z_{ij} = x^T W_{...ij} + b_{ij}$, $W \in \mathbb{R}^{d \times m \times k}$, and $b \in \mathbb{R}^{m \times k}$ are learned parameters.

The major part of deep neural network assemblage is the convolutional layers that also give its name as the convolutional neural networks mainly. The convolutional layers extract the features by assigning the convolutional neurons to local regions those calculate a dot product of the weights with the pixels values of the area. These layers include the following hyper-parameters to be optimized; zero-padding P, stride S, and the depth K and spatial extent F. The convolution layer performs the mathematical operation as given in Equation (3.17) over an image I with kernels K:

$$conv(I,K)_{xy} = \sigma(b + \sum_{i=1}^{h} \sum_{j=1}^{w} \sum_{k=1}^{d} K_{ijk} * I_{x+i-1,y+j-1,k}) \tag{3.17}$$

where σ shows the activation function, b is the bias, h, w, d represent the height, width, and depth, respectively.

The output activation volume of the convolutional layer is the feature map, where the hyper-parameters designate the patterns of it. The zero-padding is the operation of adding zeros around the image border. At first sight, it can be taught to expand the dimensions, but it enables to transmit the effects of the border elements by preserving the spatial size. Thus, the height and width of the volumes stay the same

without shrinking the height and width of the volumes, which allows constructing deeper networks. The stride how many numbers of steps will the sliding kernel window take for each application in the spatial coordinates. As a result of the kernel window stride, the spatial size reduces; however, the zero-padding adjusts it sufficiently. The stride as being the kernel shift in pixels also specifies the overlap between respective output pixels. The depth and spatial size are the numbers of kernels, each of which concentrates on different specific regions in the inputs and their dimensions in the spatial coordinates, respectively. The depth affects on the neuron count in the convolutional layer. To run a CNN model properly, the dimensions of these hyper-parameters have to be compatible with each other. In brief, there are four indispensable hyper-parameters for convolutional layers, for instance, suppose that the input volume is $W_1 \times H_1 \times D_1$ and the output volume is $W_2 \times H_2 \times D_2$, then the relations between the hyper-parameters become given as follows;

$$W_2 = \frac{(W_1 - F - 2P)}{(S+1)} \tag{3.18}$$

$$H_2 = \frac{(H_1 - F - 2P)}{(S+1)} \tag{3.19}$$

$$D_2 = K_2 \tag{3.20}$$

where W, H, and D show the width, height, and depth of the volume. Hence, it can be calculated that there occur $F \times F \times D_1$ weights for each kernel with a total of $(F \times F \times D_1) \times K$ weights and K biases due to the weight sharing property. The major stages of a convolutional layer is illustrated in Figure 3.19. The $3-channel$ input images is convolved with 3×3 kernels and then feature maps are obtained. Furthermore, the batch normalization technique [127] is worth considering in deep neural network research. However, normalization layers contribute minimally; they are useful if there exist neurons with unlimited activations, which shows characteristics very much alike to regularizers so that the regularization requirement declines. Since, the inputs of each layer of the neural network during training change, this causes the slow down in the network. For this reason, these inputs are scaled to be zero-centered with unit variance as given in Algorithm 3. The batch normalization operation increases the training speed by applying the normalization to the input mini-batches. In this way, the use of larger learning rates become feasible. In addition, the batch normalization also reduces the network sensitivity against weight initialization.

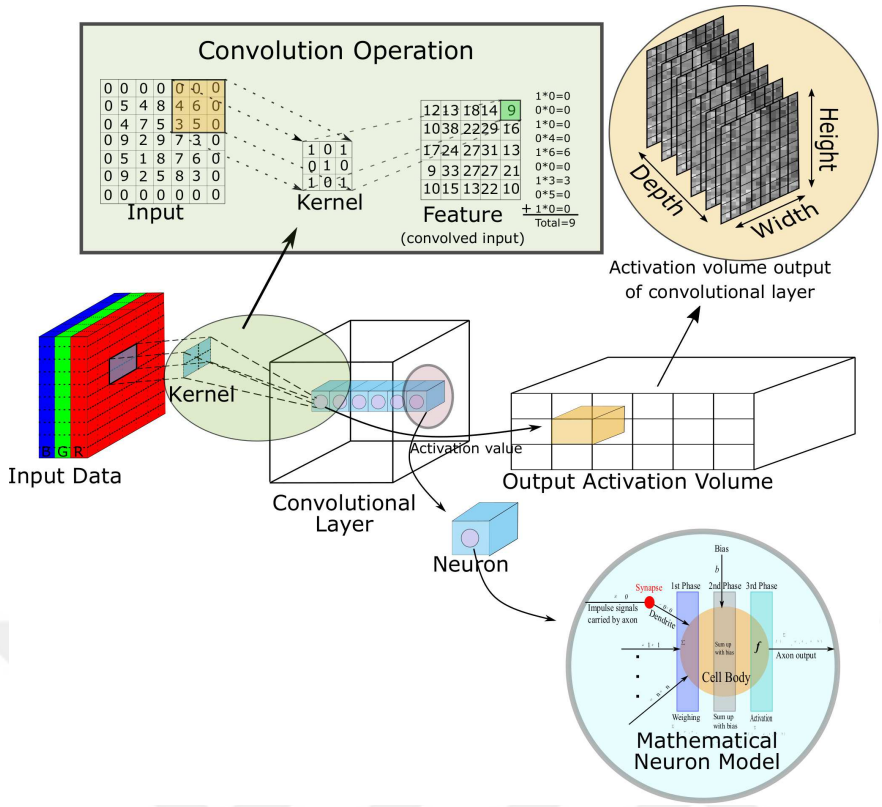


Figure 3.19 : How a convolutional layer operates over a colored input image.

Algorithm 3 Algorithm of Batch Normalization layer. [127]

1: Input: Values of x over a mini-batch from training data: $B = x_1, \ldots, x_m$;
2: Parameters to be learned: γ, β
3: Output: $y_i = BN_{\gamma,\beta}(x_i)$ // Batch Normalizing Transform.
4: Mini-batch mean: $\mu \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i$
5: Mini-batch variance: $\sigma_B^2 \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu\beta)^2$
6: Normalize: $\hat{x}_i \leftarrow \frac{x_i - \mu\beta}{\sqrt{\sigma_B^2 + \varepsilon}}$ // ε is the numerical stability constant.
7: Scale and shift: $y_i \leftarrow \gamma\hat{x}_i + \beta \equiv BN_{\gamma,\beta}(x_i)$

For each sub-network,the parameters to be learned in batch normalization procedure can be calculated by $y^{(k)} = \gamma^{(k)}\hat{x}^{(k)} + \beta^{(k)}$.

3.4.2 The smarter way of pooling techniques: the *smart-pooling*

The data size reduction (a.k.a. downsizing/downsampling or data compression) is a typical signal processing technique applied with the intention of getting rid of redundant and trivial parts of the data, but it may lead to potential wealthy information loss stored in the data. In general, the data compression can be considered in

two categories as lossy and lossless according to the reversibility situation of the downsampling due to information preservation or loss of the original data. In lossless compression, statistical models are employed to map the input signals into smaller outputs by removing the unnecessary pieces of information. This type of compression can be claimed to be reversible by remapping the output signals to the inputs with the reverse model. On the other hand, once lossy compression diagnoses the redundant parts, then eliminates it irreversibly, because there is no relation between the data elimination method and mapping.

In computer vision, data size reduction has always been an important topic that helps to diminish the quantity of data to be processed. Visual feature extraction, image downsampling, multimedia file compression, pooling layers in CNN are all such techniques works with the same ambition of easing the computations with lessening the number of data samples by disposing of them from the original data. The most crucial issue in image downsizing is to prevent valuable information loss. As anticipated, image quality degradation is an entail to the image downsizing that causes a sharper appearance utilizing lower resolution besides reducing the noise levels in the images. The spatially closer pixels are expected to collaborate to form a visual feature, so the removed pixels information has to be transferred via the pixels next to each other. The pooling is that kind of operation, which selects the pertinent pixels to map the following location in the image and it is mostly used in CNN models following the convolutional layers principally.

The pooling layer in CNNs helps to avoid over-fitting by granting an abstracted representation along with scaling down the spatial volumes and the number of parameters to learn. Another remarkable achievement of the pooling layers is that they provide geometric invariance to CNN architectures, which convolutional layers cannot handle. This layer performs on every depth slice of the former layer one-by-one and downsizes them spatially. 2×2 or 3×3 are the filter sizes utilized in the pooling layers with a stride step of 2 by a majority, otherwise, the information stored in the data is destroyed with larger receptive fields. It can be seen that the conventional settings as 2×2 kernel size and $2 - pixel$ stride for the pooling operator reduces the amount of data by %75. The following equations give the role of the hyper-parameters in the pooling layers when $W_1 \times H_1 \times D_1$ is the input volume, and $W_2 \times H_2 \times D_2$ is the

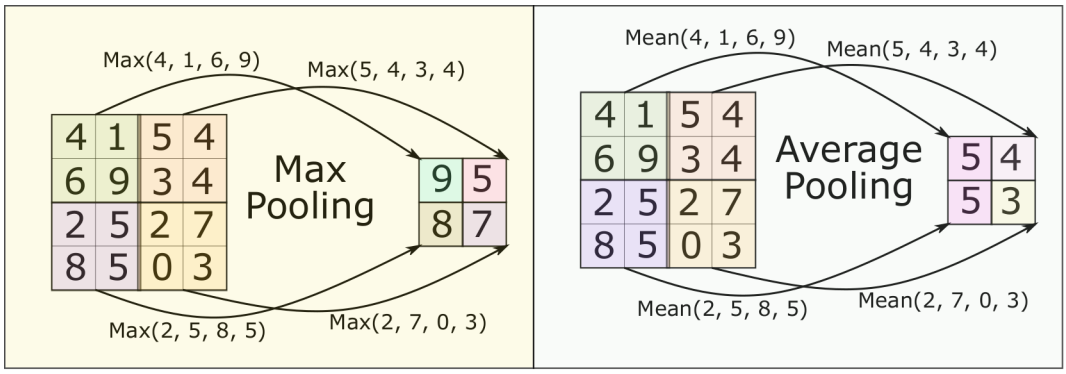


Figure 3.20 : The max-pooling and average pooling methods way of reducing the data.

mapped/output volume:

$$W_2 = \frac{(W_1 - F)}{(S+1)} \tag{3.21}$$

$$H_2 = \frac{(H_1 - F)}{(S+1)} \tag{3.22}$$

$$D_2 = D_1 \tag{3.23}$$

Although pooling layers are taught to be discarded from the CNN architectures and can be replaced by additional convolutional layers resulting in network types such as variational autoencoders (VAEs) and generative adversarial networks (GANs), they still attract the attention of the researchers with their effective implementations and acceptable performance results. Current typical usage of pooling layers relies on the selecting the maximum or average calculation within a kernel as shown in Figure 3.20 for a 4×4 input and 2×2 pooling kernel that produces a 2×2 output, but the recent focus on improving the pooling layers cannot be ignored. In [128], the performance comparison between the maximum and average pooling methods is shown as a result of various experiments conducted using conventional feature extraction methods. Once the images are downsized via one of the pooling methods, then the recognition performance of the SIFT method on downsized images is evaluated within a binary image classification task. It is shown that the discriminative power of the max-pooling is better than average pooling. Moreover, [129] states that the pooling provides invariance in feature representation as well as it may consist of spatially far-located dissimilar features. By relating the neighbor restricted pooling operation mapping ability that works for adjacent elements both in image and descriptor spaces, they claim to achieve to beat the state-of-the-art performance. Instead of identifying a pre-defined pooling scheme, [130] asserts an adaptive pooling method that learns the

receptive fields for classification tasks. Initially, several pooled features from the receptive fields are located to a spatial pyramid and then the resulting global feature vectors, which are obtained by applying Cartesian products, are classified. The tests on regular image classification pipeline show that the learning adaptive receptive fields for pooling raise the accuracy. The study in [131] suggests replacing deterministic pooling methods with a stochastic strategy. The method in this study intends to contribute to regularization of the CNN models together with being hyper-parameterless and has potential to work with other regularization techniques. It simply picks the activation from each pooling area randomly considering a multinomial dispersion. In the first step, they calculate the normalized probabilities p per region by $p_i = \frac{a_i}{\sum_{k \in R_j} a_k}$, where as are the activations within the region. Then, sampling is performed within the multinomial distribution by $s_j = a_l$, where l $P(p_1, ..., p_{|R_j|})$, a_l shows the pooled activation, l is the location from the region. Finally, the sampled activations are probabilistically weighted in each region by $s_j = \sum_{k \in R_j} p_i a_i$. They indicate their results show state-of-the-art on various benchmarking datasets with a more expensive computational cost. [132] declares a new type of pooling approach called generalized max-pooling for images classification tasks that re-weights each patch extracted by conventional methods. Alternative to bag-of-visual-words approach, generalized max-pooling equalizes the descriptor effects between the extracted patches and pooled representations besides keeping the characteristic information. Another stochastics approach to pooling layers in CNNs is introduced in [133] that employs the ordinary max-pooling and average pooling randomly during the training. It is stated that this approach helps to overcome the over-fitting. The pooled output of this method is given as $y_{kij} = \lambda \max_{(p,q) \in R_{ij}} x_{kpq} + (1 - \lambda) \frac{1}{|Rij|} \sum_{(p,q) \in R_{ij}} x_{kpq}$, where λ is a binary random number 0 or 1 that chooses the max pooling or average pooling, $|Rij|$ shows the pooling region size Rij, y_{kij} represents the pooling output of the k^{th} feature map, x_{kij} stands for the element at (p,q) pixels of the pooling region. In [134], a pooling method is explained that first extracts deep activations for feature patches and then pools them into a global vector at 3 scales by re-sampling the inputs into 256×256, 128×128, and 64×64 pixels. [135] introduces L_p units, which calculates a normalized L_p norm using the delivered signals from several unit subset projections. The normalized L_p norm is defined for given inputs $[a_1, ..., a_N]$ as; $u_j([a_1, ..., a_N]) =$

$(\frac{1}{N}\sum_{i=1}^{N}|a_i - c_i|^{p_j})^{\frac{1}{p_j}}$, where p_j is the norm order identical to per neuron, a_i are the filter outputs or input signal from lower layer, c_i shows the center or bias of the i^{th} input signal. Additionally, p_j and c_i are the learnable model parameters and so that this method differs from pre-defined structures or deterministic methods. In short, the activations of the layer below are conveyed, and then equal-sized subsets of the activations are supplied to a single L_p unit as non-overlapping sets. The geometrical meaning of the L_p unit is described by $u(x) = (\frac{1}{N})\sum_{i=1}^{N}|w_i^T x - c_i|^p)^{\frac{1}{p}}$, where w_i is the i^{th} column of the weight matrix W. Another study for the pooling layers is proposed in [136] by extracting activations from multi-scale local regions of a pre-trained CNN and then aggregating them with a scale-wise normalization to fit into the CNN model. The scheme proposed in this study inserts multi-scale pyramid pooling layer comprised of dimensionality reduction, dictionary building, activations vectors, normalization and average pooling, respectively following the replacement of FC layers by convolutional layers in the original CNN model. Afterwards, power normalization is applied to the outputs of this layers before L_1 normalization. The study suggests SVM to classify images and claims to have better accuracy rates than state-of-the-art results. A further study on stochasticity-based max-pooling operators is proposed in [137] with a fractional coefficient $\alpha \in (1,2)$ for kernel size with a degree of randomness that is related to the selection of the disjoint or overlapped pooling regions. Even though the random selection is successful on its own, it may under-fit when combined with other regularization methods such as *dropout* and data augmentation. Pseudo-random pooling region selection, which is given by $a_i = ceiling(\alpha(i+u), \quad \alpha \in (1,2), u \in (0,1))$ where a_i and u_i stand for some numbers in given range, yields more stable pooling regions as well as overlapping fractional max-pooling outperforms the disjoint one. Moreover, [138] explores to learn a suitable combination of average and max-pooling along with a tree-structured pooling operators, which is eager to learn kernels from the data, to be able to combine differentiable leaf node pooling kernels, and then agglomerate these attributes into a hierarchical tree. In addition, a method having ability to be responsive that is maintained by a *gate* is also developed. These three approaches are employed within existing CNN models together with other regularization techniques, and it is claimed

that all of three pooling strategies (mixed, gated and tree-based) raise the accuracy rates and *tree+max-average* pooling configuration results in state-of-the-art performance.

Biological roots of the spatial pooling are primitively associated to the visual cortex in [139], which states that mid-level features are invariant to small deviations. Similarly, [140] examines the structure of receptive fields in the visual cortex and reports that non-linearity emerges from the spatial pooling. In [141], a new kind of spatial pooling is highlighted that focuses on essential regions to transfer the extracted features carrying discriminative representations. In this thesis, we declare a new kind of pooling strategy called *"smart-pooling"* that performs the actions given as Algorithm 4.

Algorithm 4 Smart-Pooling scheme.

Require: Spatial kernel size $N \times N$ (2×2 by default, otherwise 3×3),
stride S (2 by default, otherwise overlapping or sparse coding.),
zero-padding P (fulfils to keep the input size suitable for pooling by default.).

i) **Input:** Image or previous layer pixels.

ii) Arithmetic mean of the kernel: $M = \frac{1}{N \times N} \sum_{i,j=1}^{N} a_{ij}$, where a_{ij} shows the pixels values and i, j are the pixel indices inside the kernel.

iii) The pixels higher than the mean: $H = append(a_{ij}) > M$. $N(H)$; count the elements of H.

iv) The pixels smaller than the mean: $L = append(a_{ij}) < M$. $N(L)$; count the elements of L.

v) Compute ***Smart-Pooling*** output PL as follows;

if $N(H) > N(L)$ **then**
$PL = \frac{1}{N(H)} \sum_{i,j=1}^{N(H)} H_{ij}$
else if $N(H) < N(L)$ **then**
$PL = \frac{1}{N(L)} \sum_{i,j=1}^{N(L)} L_{ij}$
else
$PL = M$
end if

The toy example illustrating the difference between *smart-pooling* and average and max-pooling given Figure 3.21 that it operates with the objective to reflect the distinction between input elements ideally to the outputs by converting them more salient. The *smart-pooling* is a competitive algorithm to the average and max-pooling techniques concerning computational efficiency, preventing over-fitting, and carrying semantic information within the kernels to the next layers in both back and forth directions. Conjointly, smart-pooling provides robustness to noise and contributes to regularization performance of the model as well. As one can see from the procedure

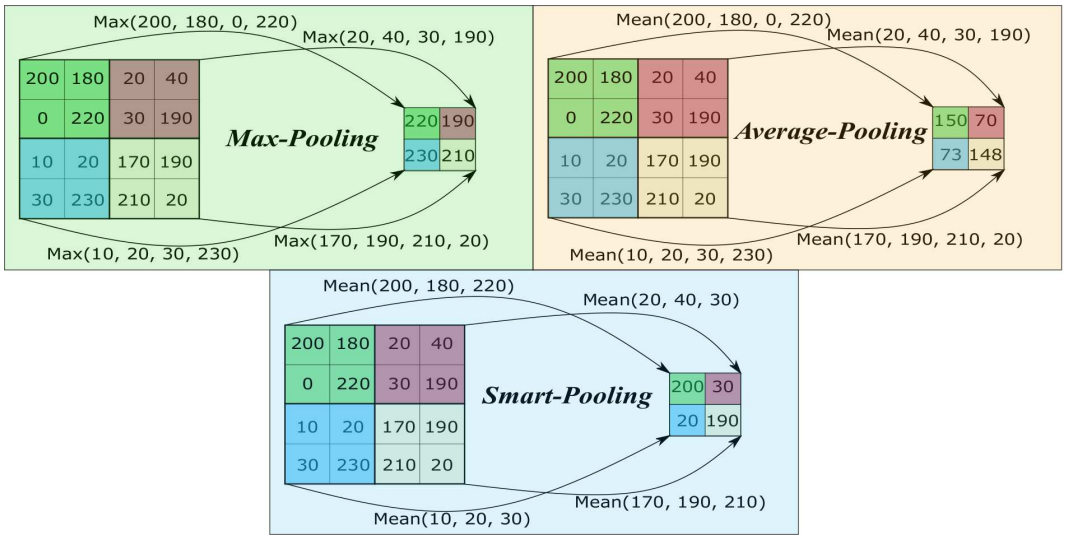


Figure 3.21 : The *smart-pooling* example the besides max-pooling and average pooling.

of the *smart-pooling* that it has the ability to behave both average pooling and max-pooling likewise. But, it can generate different outputs than those as well. In addition, *smart-pooling* guarantees that the performance will be at least as the average pooling level. On the one hand, *smart-pooling* can eliminate the outliers or noisy pixel inside the kernel and does not take into account. On the other hand, it only focuses on the majority of the pixels that brings computational efficiency and better generalization performance at the same time.

3.5 Results

Convolutional neural network architectures with a focus on object detection and recognition are explained in detail, besides a new image dataset *ADORESet* is introduced and a competitive pooling algorithm is asserted thus far. In this section, after presenting the object recognition performance results of various CNN architectures, we will give the object detection performance of *RetinaNet*, and the effects of *smart-pooling* by comparing with the average and max-pooling methods is also shown. The experiments of object recognition and detection tasks are conducted on a computer that runs a Linux distribution of $64-bit$ Ubuntu 14.04 equipped with an NVIDIA GTX 1080 GPU, an Intel $i7$ CPU $920@2.67GHz \times 8$, $6GB$ RAM and $1TB$ hard-drive spins at $7200RPM$. The comparisons of pooling methods are performed on a laptop computer, which operates a Linux distribution of $64-bit$ Ubuntu 14.04 rigged with an NVIDIA GTX $930M$ GPU, an Intel $i7-6500U$ CPU $@3.10GHz \times 8$, $8GB$

Table 3.2 : Data configurations for experiments using *ADORESet* including data types and number of images.

Type of Training Data	# Images	Type of Validation Data	# Images
Real Images	1775	Real Images	725
Real Images	2000	Real + Synthetic Images	500 + 500
Real Images	1500	Synthetic Images	750
Synthetic Images	750	Real Images	375
Synthetic Images	600	Real + Synthetic Images	150 + 150
Synthetic Images	500	Synthetic Images	250
Real + Synthetic Images	750 + 750	Real Images	750
Real + Synthetic Images	1775 + 500	Real + Synthetic Images	925 + 250
Real + Synthetic Images	375 + 375	Synthetic Images	375

RAM and $1TB$ hard-drive spins at $5400RPM$. Equally important as the hardware, we used Python language as the software and Keras [142] wrapper with a Tensorflow [143] backend is chosen to develop the object recognition and localization algorithms.

3.5.1 Object recognition performance

The way for detecting and recognizing objects in deep neural networks is through training for many times with sufficient amount of data until reaching predefined performance criteria. To reveal the effects of the hybrid dataset on object recognition task comprehensively, we give performance results of all possible combinations of real and synthetic images as being training and validation data. These combinations with regard to the types of data for training and validation with the number of images are given in Table 3.2. Hence, 36 performance results are obtained for 9 data content formats and 4 deep CNN methods in terms of time, accuracy and loss values. The number of frozen layers, which are kept the same as weights of base models, of deep CNNs ([78], [81], [79], [82]) are varied depending on the number of data. The training epoch number is fixed to 50, which ensures the convergence of performance measures to stable values. ReLU is chosen as activation function, which is used for all configurations. Stochastic gradient descent [144] is used as optimization method while fine-tuning the [78] model and Adam [145] is used for the remaining models. To calculate the probability of the output in the classification layer, the softmax function

is applied at all models. The batch size is varied with respect to the memory capacity of the system.

3.5.1.1 Wild web images as training data for object recognition

Solely real images as training data and combinations of real and synthetic images as validation data are used in the first 3 of the experiments as details are given in Table 3.3. The progress of accuracy and loss values throughout 50 epochs of training and validation are also displayed in Figure 3.22. As can be seen from both Table 3.3 and Figure 3.22 that the highest validation accuracy rates are achieved when the real images are used for training and validation. *Inception-v3* is slightly better regarding the validation accuracy than the other models while *VGGNet* is trained in the shortest period. The batch size of all configurations is set to 32 except the case that the real and synthetic images are used for validation by *Xception* model because of memory issue, which is handled by setting the batch size to 16 for this configuration. Training accuracy values for all methods in all data pair cases give acceptable results at around %95 but not the validation accuracy values. As can be seen from both Table 3.3 and Figure 3.22 A) that similar training and validation data types result in high accuracy rates for all models. Nevertheless, usage of incompatible data pair yields unsatisfactory validation accuracy rates. Such that using a mixed type of data presented in Figure 3.22 B) when training data consists of only real images yields approximately %50 as validation accuracy rate. Moreover, validation accuracies of all models fluctuate around %10 in the worst case, which is displayed in Figure 3.22 C), the real images are used for training and synthetically generated images are used for validation.

3.5.1.2 Simulation environment images as training data for object recognition

If only the synthetic images are fed into the CNN models as training data while validation data is varied, the performance parameters form as displayed in Table 3.4. The progress during the training and validation sessions are given in Figure 3.23. Similar to previous results, data types show discriminative characteristics concerning the performance metrics. The batch size values for all cases are set to 32. The validation accuracies for the example of having the same data types for both training and validation are the highest in all cases. The decrease in validation accuracy rates is distinct when the real images are supplied to the model as validation data. One can

Table 3.3 : Performance results if training data consists of only real images. (R stands for real images and S stands for simulation images. The numbers near R and S denote the number of images.)

Model	Data Type and Amount		Train	Val.	Time per	Batch
	Train	Validation	Acc (%)	Acc (%)	Epoch (sec)	Size
VGGNet	R 1775	R 725	84.82	80.44	435.40	32
	R 2000	R 500 + S 500	98.32	50.86	660.66	32
	R 1500	S 750	98.23	9.30	483.88	32
InceptionV3	R 1775	R 725	96.81	86.54	1634.7	32
	R 2000	R 500 + S 500	97.17	50.97	2657.4	32
	R 1500	S 750	98.23	10.77	872.16	32
ResNet	R 1750	R 750	97.00	86.01	472.9	32
	R 2000	R 500 + S 500	97.72	49.54	1094.2	32
	R 1500	S 750	97.85	7.87	415.02	32
Xception	R 1775	R 725	97.44	85.64	1706.50	32
	R 2000	R 500 + S 500	97.67	49.46	2667.60	16
	R 1500	S 750	97.61	7.04	1974.90	32

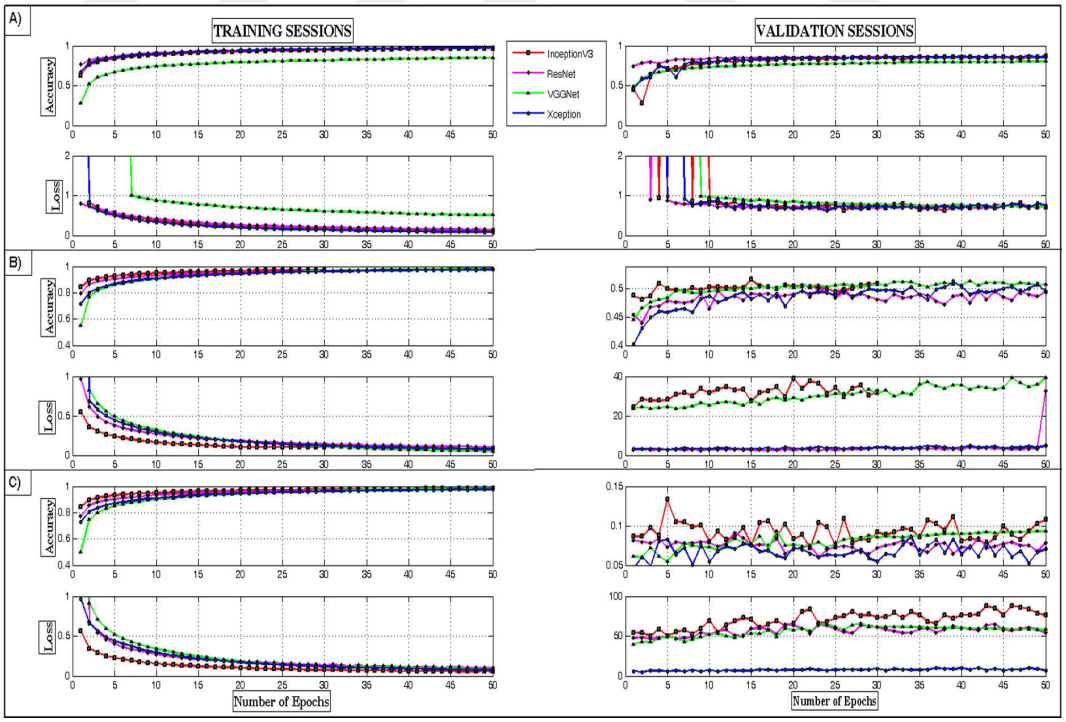


Figure 3.22 : Progress of performance parameters during training and validation sessions. Training data is composed of only real images. A) Real images for validation B) Real and simulation images for validation C) Simulation images for validation.

Table 3.4 : Performance results if training data consists of only simulation images. (R stands for real images and S stands for simulation images. The numbers near R and S denote the number of images.)

Model	Data Type and Amount		Train	Val.	Time per	Batch
	Train	Validation	Acc (%)	Acc (%)	Epoch (sec)	Size
VGGNet	S 750	R 375	98.87	5.01	185.92	32
	S 600	R 150 + S 150	98.11	53.80	175.56	32
	S 500	S 250	97.03	95.92	164.44	32
InceptionV3	S 750	R 375	89.93	5.71	492.53	32
	S 600	R 150 + S 150	97.61	51.63	484.94	32
	S 500	S 250	98.78	97.58	475.64	32
ResNet	S 750	R 375	93.49	7.85	299.37	32
	S 600	R 150 + S 150	96.13	49.41	284.41	32
	S 500	S 250	98.49	95.53	275.82	32
Xception	S 750	R 375	97.37	5.05	745.85	32
	S 600	R 150 + S 150	96.91	47.00	687.17	32
	S 500	S 250	97.67	95.27	666.00	32

easily say that the data type incompatibility is explicit as resulting in the lowest rates when the data type configuration is set to utilize synthetic images as the training data and real images as validation, as can be seen from both Table 3.4 and Figure 3.23 regarding all model outputs. In other words, the variations of synthetically generated images are not able to cope with any case if the real images are used for validation.

3.5.1.3 Hybrid data images as training data for object recognition

In this study, only one single type of images are adopted as the training data so far. From this point on, a various number of hybrid data depending on the validation data type is fed into the models as the training data. Additionally, the total quantity of training and validation images are the largest for the hybrid training data type case. Subsequently, the time spent during the operations is the longest as can be seen from Table 3.5 with other performance outputs such as accuracy scores. All fine-tuned models succeed in surpassing the results of base models by exploiting the real and synthetic images commonly as training data where the session performances are showed up in Figure 3.24. The batch size for all models is adjusted to 32 other than the cases of real and real-synthetic images as validation data combinations for *Xception* model, which are fixed to 16. Thus, the memory requirement of *Xception* is higher than other models that depend on the number of layers updated during the fine-tuning and natural structure of the model itself. As a result of these performance

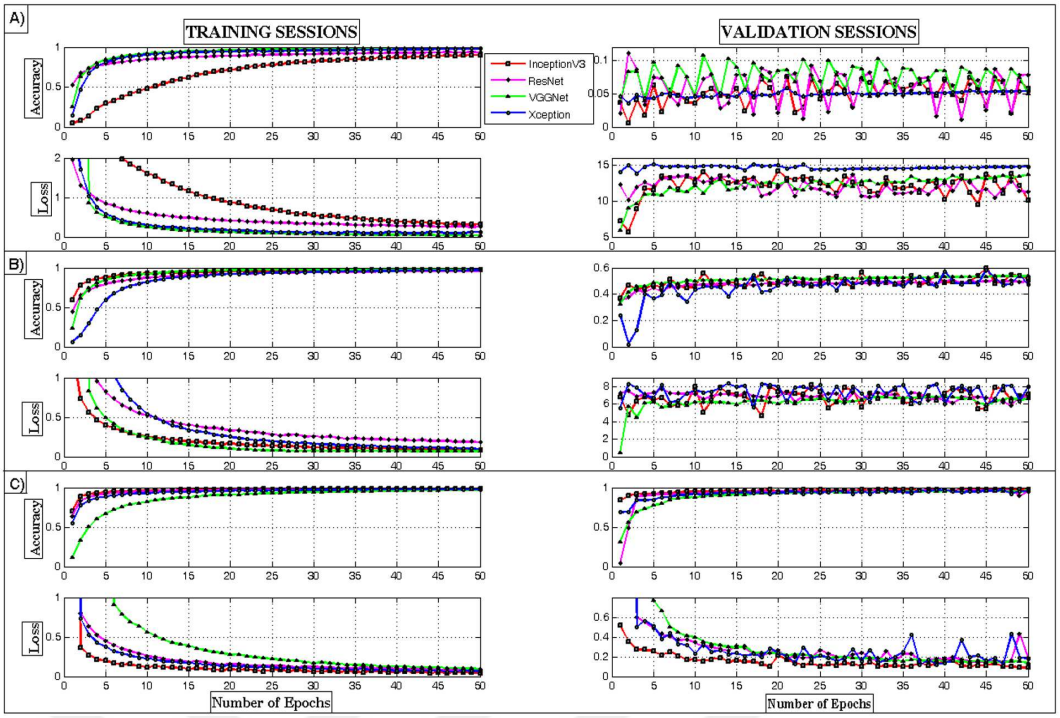


Figure 3.23 : Progress of performance parameters during training and validation sessions. Training data is composed of only real images. A) Real images for validation B) Real and simulation images for validation C) Simulation images for validation.

appraisals, the hybrid format of *ADORESet* is proved to be robust to the type validation data.

3.5.2 Object detection performance

In this thesis, the object localization through detection is defined to specify the bounding-box locations that surround the objects of interest within the whole images. In this part, we present the outputs for the experiments conducted towards regressing the bounding-box locations around the objects by adopting the *RetinaNet* with *ResNet-50-FPN* architecture. It is worth to note that the *RetinaNet* is variant of $50-$*layered ResNet* and *FPN*. To train this model according to the our object detection consideration, we cancelled the object recognition branch that is one of the sub-models remaining after the feature pyramid network of the original architecture. Thus, we reduce the quantity of the parameters to be trained. We fixed the batch-size to 8 and the number of steps for each epoch to 10000. The number of epochs is set to 30 that is sufficient to converge the satisfied performance levels. The training is executed by employing 13000 images in 5 object categories from *ADORESet*, which are bottle, can,

Table 3.5 : Performance results if training data consists of both real and simulation images. (R stands for real images and S stands for simulation images. The numbers near R and S denote the number of images.)

Model	Data Type and Amount		Train	Val.	Time per	Batch
	Train	Validation	Acc (%)	Acc (%)	Epoch (sec)	Size
VGGNet	R 750 + S 750	R 750	95.49	85.37	427.96	32
	R 1775 + S 500	R 925 + S 250	98.08	90.50	717.18	32
	R 375 + S 375	S 375	96.48	93.06	194.2	32
InceptionV3	R 750 + S 750	R 750	96.54	86.03	495.85	32
	R 1775 + S 500	R 925 + S 250	98.15	89.97	1685.51	32
	R 375 + S 375	S 375	95.76	93.54	432.26	32
ResNet	R 750 + S 750	R 750	95.88	86.70	427.64	32
	R 1775 + S 500	R 925 + S 250	97.02	87.54	609.44	32
	R 375 + S 375	S 375	95.05	91.60	212.72	32
Xception	R 750 + S 750	R 750	99.54	90.41	497.44	16
	R 1775 + S 500	R 925 + S 250	97.74	89.00	2408.61	16
	R 375 + S 375	S 375	98.01	96.27	645.00	32

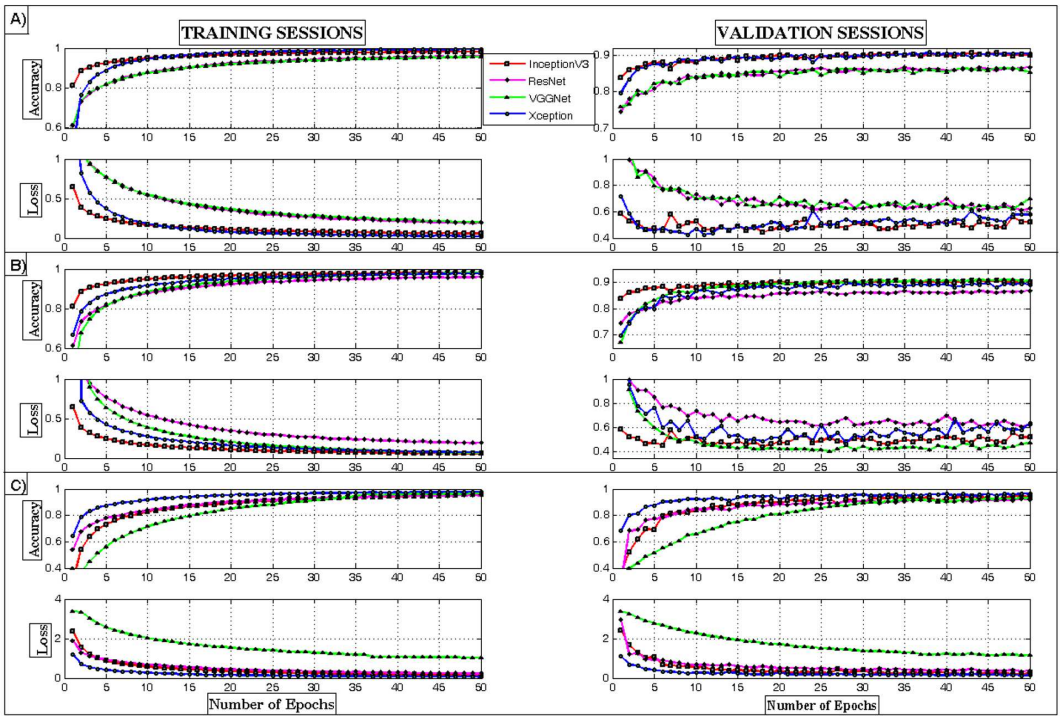


Figure 3.24 : Progress of performance parameters during training and validation sessions. Training data is composed of both real and synthetic images. A) Real images for validation B) Real and simulation images for validation C) Simulation images for validation.

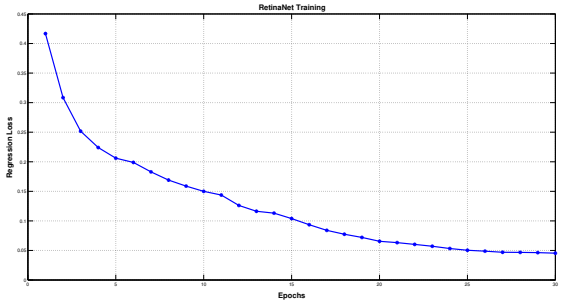


Figure 3.25 : Object detection performance as of regression loss.

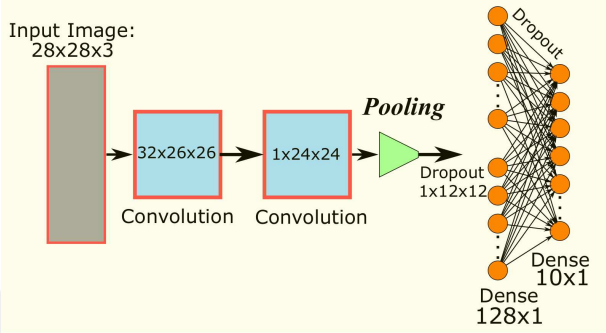


Figure 3.26 : The model trained to compare the pooling methods.

cup, speaker, and vase. The main reason why we chose these objects is to decrease the training time as well as we will manipulate these objects in the case study. The average time spent during training per epoch is 7754s. We test the model on a dataset composed of 500 images for each class and the response time for a query image of this model is 254ms. During the tests, we adjusted the IoU to 0.5. In other words, if the estimated bounding-box overlaps the ground-truth by %50, then we count it as a successful output. The progress of the regression loss during the training is displayed in Figure 3.25. After 30 epochs, the regression loss does not fluctuate and stays persistent under 0.050 and the resulting output is 0.0454.

3.5.3 Smart-pooling performance

This study attempts to extend the current literature with an innovative pooling approach called *smart-pooling*. The *smart-pooling* provides a transitive structure composed of average and max-pooling methods as well as it hosts the advantageous behaviors from both of them. To evaluate the working principle and measure the performance according to the average and max-pooling methods, we tested the *smart-pooling* on MNIST [74] dataset, which is composed of 60000 training and 10000 test images of handwritten digits in 10 categories, and the adopted model is displayed in Figure 3.26.

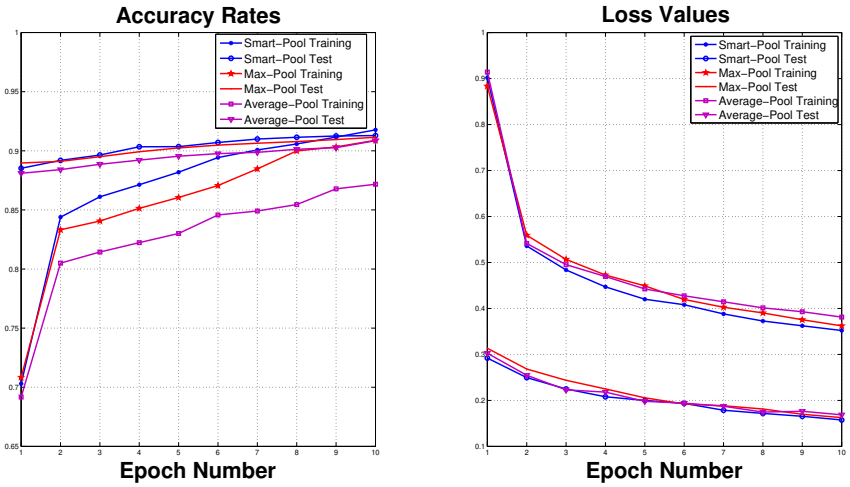


Figure 3.27 : The model outputs as of accuracy and loss values according to the change in the pooling method.

The configuration of the CNN model is as follows; activation function is ReLU and the prediction is executed by softmax, AdaDelta [146] is the optimization method and the loss function is categorical cross entropy. The batch-size is set to 128 and the training is performed for 10 epochs. The training and testing performance results can be seen from Figure 3.27.

The accuracy rates and loss values prove that the *smart-pooling* strategy quantitatively surpasses both average and max-pooling methods. The test accuracy of the model with *smart-pooling* is %91.29 while the accuracy of the model with average and max-pooling are %90.86 and %91.14, respectively. The test loss of the network with *smart-pooling* is 0.1556 while the loss of the network with average and max-pooling are 0.1623 and 0.1687, respectively. The reason of these values appear to seem worse than the original models that we modified the model to reduce the computational time with acceptable results. Moreover, the *smart-pooling* also outperforms the remaining methods regarding training time by 21s per epoch while the average pooling training per epoch takes 24s and the max-pooling training per epoch takes 22s. Even though these measurements are useful, the complete qualification of such an algorithm has to be measured qualitatively to give insight about the working strategy of the method. Therefore, we applied these three pooling techniques to an image shown in Figure 3.28. Initially, the image has the dimension of $300 \times 300 \times 3$ and it is downsized to $150 \times 150 \times 3$. In other words, the image is rescaled by a factor of $\frac{1}{4}$. Although the outputs only have the size of %25 of the original input image, it is clear that the *smart-pooling*

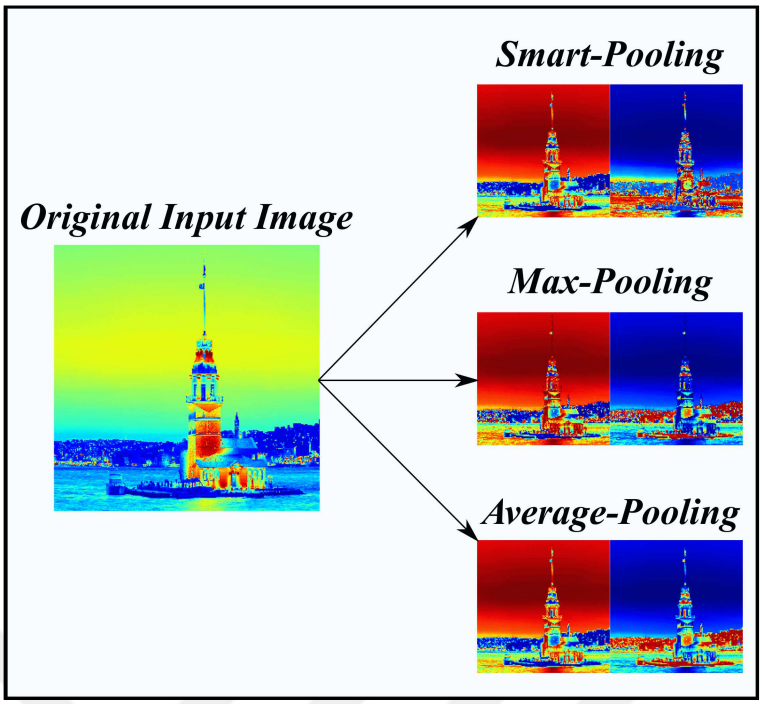


Figure 3.28 : Pooling methods comparison on image sub-sampling.

output reflect pretty much all the details concerning the color transitions and primitive image features such as edges and corners. Furthermore, the *smart-pooling* output image contains more salient features that yield more comprehensible and reasonable consequences. For instance, the flag-tower is almost to imperceptible in the output images of other methods.

3.6 Discussion and Conclusion

Object detection and recognition for robotics research in the context of dexterous manipulation, grasping, tracking are still challenging hot research topics. However classical computer vision and control methods proposed successful solutions, deep learning based methods outperformed them with the support of software and hardware developments, which enabled to run such deep neural networks in feasible periods. Therefore, the demand towards different types of datasets, as being the most decisive part of learning systems, is increased. Whether labeled or unlabeled image datasets with millions of images for thousands of categories exist, parameters such as a number of images per category, image types and formats, object classes, dimensions, etc. play an important role to select a dataset. From this point of view, *ADORESet* with its versatile hybrid structure allows researchers to implement their algorithms both for

real-world and simulation environment conditions. Additionally, *ITUrk* GUI makes it viable to label, eliminate and resize massive amount of images. Furthermore, relationships between object categories are presented using the annotations of the successor objects. To the best of our knowledge, our study provides one of the most comprehensively detailed experimental performance results for state-of-the-art CNNs, besides a new densely labeled hybrid dataset. Despite the fact that the incompatible data pairs yield useless deep CNN weights for all models, the performance results reveal that usage of real and simulation images together as training data gives satisfactory validation accuracy rates whatever the validation data is.

In this thesis, we refer to only category or label prediction by object recognition and the bounding-box regression by the term object detection. As a result of our object recognition experiments, we choose the *VGGNet* architecture for the remaining parts of this thesis involving object recognition tasks. The VGGNet excels amongst other regarding the accuracy, robustness and it is very easy to code due to its architectural simplicity. On the other hand, *RetinaNet* is the only model experimented for object detection as long as it exceeds the other methods with its one-stage procedure and robustness. It is also remarkable that the *smart-pooling* is superior to the mostly used average and max-pooling methods in terms of qualitative and quantitative performance measurements.

It has to be underlined that our reproducible results prove the significant power of training-validation data types. In essence, once a CNN model is obtained as a result of training hybrid dataset such as *ADORESet*, then it can be applied to real and simulation images together or separately. Additionally, *ADORESet* is suitable for developing novel algorithms, which can be CNNs or classical methods, intended to detect and/or recognize objects. These applications aspire to include *ADORESet* and CNN based recognition to augment better grasping and manipulation performance in service robotics. The smart-pooling technique has also potential to serve designing innovative CNN architectures with its compact and competent structure incorporating the favorable properties of the most common methods.

4. CASE STUDY: THE OBJECT MANIPULATION EMPLOYING A ROBOTIC MECHANISM

This study was conducted in the form of a series of experiments, with the data from newly presented image dataset, in the manner of object recognition, object localization adopting both conventional methods and deep learning techniques in conjunction with introducing *ADORESet* and a pooling strategy called *smart-pooling*. The outcomes are also analyzed both qualitatively and quantitatively. In fact, the methods utilized in the manipulation scenario are determined together with the relevant objects. This chapter takes the form of integrated simulation and empirical scenarios involving similar robotic mechanisms that manipulate the visually recognized objects standing on a table surface called *"Deep Table"*, because it is equipped with the cameras, robotic mechanism, and mechanic structure. The *"Deep Table"* has similar properties in simulation and real-world cases.

4.1 Introduction

Vision is an essential component in humanoid robotics and plays a key role in developing perceptual systems. When vision is the case for intelligent algorithms, then the datasets are the primary issues concerning their compatibilities to the particular applications. Moreover, the content of the data samples is as much important as the adopted algorithms. Following the dataset arrangement, the routine machine learning actions become the main considerations for a successful implementation. For instance, a robotic application may only require extracting semantic information stored in images and count the objects belonging to a specific category while another implies to move to place the objects in relevant locations. These details define the framework of the learning system. There ensure blurred boundaries surrounding the smart systems nevertheless the limits are rigorously defined in the deterministic approach. In other words, the smart mechanisms can favorably proceed and settle when an unexpected or unforeseen disturbance or system input arises; however, the deterministic systems fail in such situations. Artificial intelligence by means of deep neural networks is

increasingly set to become a vital factor in robotic systems. A striking aspect of DNNs is that the prevalence in complex systems together with the simplicity of implementation. In most cases, DNNs are able to produce ready-to-use models in different domains and platforms. As a result of proving its human-level performance scores in different areas, DNNs conquered the territory, and most of the recent works incorporate it. Thus, the DNN-free attitudes are assumed to be obsolete as long as the system is not able to infer from or respond to inputs successfully. In contemporary robotics approach, it is generally admitted that the DNN-based algorithms employ the sensory data and generates meaningful semantic decisions and/or actions as well as the robotics mechanisms have been converted to more skillful forms, which enable them to succeed in complicated tasks, i.e., variable stiffness joints.

A striking property of DNN models is to have the capability of processing big data and achieving human-level performance results. Since this thesis covers the manipulation of visually recognized objects, we focus on the visual applications of DNN models. For this reason, we retrained the most prominent CNN architectures to recognize the objects adopting *ADORESet* along with to localize the objects within scenes, namely we fine-tuned these pre-trained models with our custom dataset. In consequence, we configured the visual system of our case scenarios composing of object recognition and localization with the implemented *VGGNet* and *RetinaNet* architectures, respectively. Moreover, the modern robotics needs further processes for the outputs of these CNN models to extract semantic information, e.g., the relation between the recognized objects, the behavior of these objects, their physical attributes. In *ADORESet*, we provide the information that belongs to the object relation depending on the appearances within the same image. Beyond that, we also assigned identical physical properties to each object considered to be manipulated, and then we built our control procedure on top of the semantic information both for simulation and experiment sessions.

It is often stated that the primary objective of robotics researches is to mimic human behavior and motions favorably; however, the existent view beneath this is to beat the human-level performances because of up-to-date demands, necessities, and requirements. The importance of robotic mechanisms is still poorly understood in the context of their human imitating potentials. Even if the robots serve in the

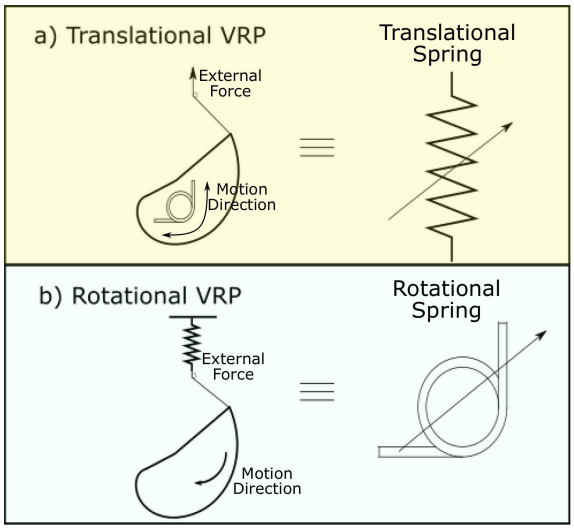


Figure 4.1 : Variable radius pulley types a) Translational VRP, b) Rotational VRP.

industry with satisfactory results for many years within strictly restricted workspaces and missions, the advancement in humanoid robotics still is not near to reach that level. Therefore, variable radius pulley (VRP) mechanisms are asserted to provide the recent expectations regarding stiffness arrangement, zero backlash, gravity compensation, simplicity, and compactness. The stiffness adjustment ability is maintained by the springs and allows the robots to work appropriately under different conditions. The springs can be inserted into a VRP system in two ways as shown in Figure 4.1 a) and b) as translational spring placed in the middle of the pulley and rotational spring placed outside the pulley, respectively, where a cable wraps around the pulley profile. The pulley radius shift induces non-linear moment-rotation affair. These force/moment-deflection specifications are provided by the pulley profiles. Therefore, we intend to direct one of the innovative approaches to close the gap between desired and existing performance levels and adopt a variable radius pulley-variable stiffness joint (VRP-VSJ) mechanism to execute the manipulation task properly. In this way, our mechanism can adjust the stiffness and forces to be applied distinctively to the recognized objects according to the properties.

This chapter is organized as follows; the literature review is given in Section 4.2, and then the simulations are provided in Section 4.3. Afterwards, in Section 4.4, the tests on the robotic mechanism are conducted. In the last part of this chapter, the results are revealed before the conclusion and discussion in Section 4.5 and 4.6, respectively.

4.2 Literature

A considerable volume of literature has been published on the robot-object interaction strategies. What is more, the properties stored in objects and talented mechanisms are neglected because of the lack of the computational power and reluctant attitude to VRP-VSJ mechanisms. In traditional rigid robot designs, external force/torque (f/t) is estimated using motor currents as explained in [147] or frictional model identification as given in [148]. All of these methods demand initial off-line calibration or identification processes. The rigid mechanisms cannot nevertheless afford the requisite which dictates the variation in stiffness for modern robotic applications working under different conditions. Reasonably, the VSJs are remarkably better choice than conventional systems, especially considering the dominance in the energy consumption and force interaction.

Flexible joint mechanisms have compliance that facilitates to compensate f/t passively. The study in [149], declares an antagonistic cable-driven mechanism using two motors for angle and stiffness adjustment separately to approximate human-like motion. A major part of the robotic tasks, social robotics in particular, necessitate safe force interaction. The change in the muscle stiffness is stated as linearly implying a quadratic stiffness characteristic from the spring. Moreover, for more complex force-deflection necessities of robots, [149] proposes a cam mechanism imitating the biological muscle motion with quadratic properties. The tests are run in an antagonistic setup, and linear stiffness characteristics are observed by co-contraction. Alternatively, variable radius or non-circular profiles in pulley designs can be used both in translational [150] and rotational [151] ways. In [151], to produce a non-linear force-torque deflection profile, a torsional spring is utilized unlike the VRP mechanisms adopting translational springs to generate spring elongation based torque profile. The readers seeking for more information on this topic, should refer to read [152], to which partially contributed as a part of this thesis, that declares the VRP-VSJ mechanism employed in this thesis to perform the manipulation task in the way of approximating the human approach.

One of the most important engineering considerations is to minimize the cost. In this study, we aim to estimate the external f/t affecting onto the objects to be manipulated following the visual recognition and localization which is obtained without additional sensors rather than encoders. The dynamic working conditions

make it more difficult for conventional model-based estimation of external f/t effecting on the joints because of the uncertainties and errors due to the lack of sufficient mapping capabilities real-world conditions to the mathematical models. The study in [153] reveals the success of ANNs to handle non-linearities in dynamical systems. It is common for model-based estimation methods to linearize non-linear systems by making assumptions and then perform the algorithm with a linear observer [154]. Although the performance of ANNs depends on data regarding many factors such as size, diversity, reliability, etc., as shown in [155] they give successful tracking results using a 2 DoF robotic arm when there is uncertainty, which makes them preferable for real-world applications. Likewise, [156] trains a neural network with two hidden layers to estimate the payload of a series manipulator using trajectory data for both single joint and multi-joint cases. While learning to estimate external forces is still a challenging problem in robotics, [157] uses convolution based deep neural network, which is trained by sequences of depth images as point clouds, to estimate robotic arm motion. For estimating contact force in haptic applications without expensive devices, [158] introduces ANN-based force/torque observers using only position and torque data. Beyond acceptable force/torque estimation, tracking and control instances of ANNs using simulation and/or measured data from various sensors in robotic systems, our study proposes to estimate external force/torque values of a VRP-VSJ mechanism using solely measured encoder data with ANN models.

In unknown environments, when the robot interacts with the objects inside without preconfiguration, then external sensor usage becomes compulsory to execute the tasks. Even if the visual data is sent to the system as feedback, semantic knowledge may raise the sensitivity and helps the robot to approximate human-level manipulation motion. The particular areas of the way people interact with objects and how the things affect on this still remain obscure; however, a noticeable quantity of studies are being carried on. On behalf of clarifying the neurophysiological background of visual perception and its relation with actions, [159] emphasizes that the visual perception generates intention which belongs to the observed subject. The same study suggests that the natural motion recognition is executed through the curing by the perception and human motor functions. There also exists a connection between perception and motor control domains that interacts and affects each other. Additionally, the extended analysis

exposes the neuroimaging results of primate cerebral cortex that the intention of the aim influences the visual pathway status, for instance, it is consistent with biological motion if the aim is explicit; otherwise, it is implicated. The study in [160] investigates the working principles of the brain concerning the sensory inputs effects on the actions by evaluating an experiment conducted on astronauts who estimate time-to-contact with oncoming objects. It is indicated that the brain trusts a second-order internal physical model and the intuitions prevail, even though the objects are visually seen, and the astronauts know the conditions. However, time-to-contact is not successfully estimated in the zero-gravity situation, and an internal gravity model interferes the actions despite the explicit vision. [161] highlights how people perform object grasping and manipulation and what lies beneath the decision of the way executing these tasks by observing the velocity and force transmission ellipsoid motions. The primitive relations of object affordances with human understanding and consideration of them as grasping and manipulation tools are explained in detail. Moreover, 7 people participated in the tests to perform 9 grasping motion with 5 objects at their fingertips. The people wear a glove equipped with sensors and orientation, velocity, position, and stiffness data are gathered. The findings reveal that the grasp stiffness depends on the situation of the hand and conducting more fingers during grasps raises the stiffness. So that the anticipated grasp schedule is not only performed according to the object geometry, but also the desired manipulation. In addition, the finger locations on the objects change the velocity and force transmission ellipsoids. In [162], the importance of actions in object recognition is examined empirically whether visual object recognition and object-directed actions are separate processes happening in primary visual cortex and dorsal visual system, respectively. It is indicated that there is not an exact clue on the relationship between action representations and visual object recognition; however, there are remarkable evidences on the interactions between object recognition and object-directed actions that also influence each other. The first experiment in this study is conducted employing the known objects, which are shown under poor light conditions, and the action priming is observed concerning the accordance with motor interactions. In the second experiment, the constitution of action priming is monitored with the intention of revealing the relationships with action representations depending on object information and the meaning of objects. The existence of the priming effect on naming accuracy for precedence of different objects

with similar motor interactions and the absence of the action priming effect for verbal stimuli are proposed as outputs of the experiments, respectively. In other words, the effect of visual information of objects is more important for human actions rather than the verbal meaning of the objects. [163] investigates the function of motor actions in visual perceptual evaluation by testing the recognition, association, and manipulation motions for the same objects on 37 unilateral stroke patients. The findings assert that there is interrelationship between the usage of objects and their recognition as well as there is connection between the motion recognition and replication. However, they claim the motor actions do not provide apparent meaningful information to conceptual framework directly.

Unlike neuropsychology-based studies, [164] refers to analyze the human-objects interactions over visual data with a Bayesian approach that incorporates the perceptual knowledge with the object response to manipulation or similar actions. The study explains two models to interpret human-object interactions in videos and static images by fusing the action and object recognition in the same framework. Moreover, it is proposed that the recognition performance is raised due to the combination of functional and spatial context. [165] proposes an autonomous task dependent grasping method for visually recognized objects by using conventional feature matching methods. The geometry of objects is taken into account for the humanoid robot as experimental platform and the grasps are performed with the visual servoing approach. The object categorization process is comprised of segmenting out the objects, recognition, and pose estimation while the grasping schema is composed of the perception (segmentation, recognition, and orientation arrangement), prediction (grasp hypothesis generation, task identification, and grasp method selection), and action (visual servoing and tracking the object and marker) steps. The results of this study claim to increase the efficiency and success rates of the grasp tasks for the autonomous motions; however, the tasks and motion characteristics depending on the object categories are predefined and given to the humanoid robot. In a simlar manner, [166] examines the behavior transfer approaches regarding impedance from people to robots by imitation learning. To generate the desired motion with variable impedance actuators (VIAs), they investigate whether it is more propriety of the direct transfer of human impedance symptoms to a correspondent robotic arm or

scan the values minimizing the cost function applying an inverse optimal control. The findings suggest that feature-based tracking is appropriate for supervised learning from human teachers as long as the inverse optimal control is better if the behavior will be transferred from a robotic system to another. The behaviors of 21 human subjects for reaching or slicing motions are examined in [167] within an imposed visual feedback delay sequence to verify if the brain can handle the variations in a feedback loop delay and rearrange itself. The effects of visual feedback on human motor control functions are argued similar to [160], and the overshoot of the motions are indicated as the evidence of vision-based feedback control in humans. However, after several trials, the human subjects readjusted themselves to the new situation successfully. In the last part of this experiment, the delay is removed instantly, and the subjects undershot the target this time. Furthermore, in the second test in this study, the same findings are obtained for harmonic back-and-forth motions. The relationship between the feedback and feedforward controllers are also examined, and it is stated that these controllers collaborate with visual feedback while performing movements and motor learning. [168] proposes a model involving an anticipatory temporal conditional random field with the aim of predicting the next human motion sequences. The object affordances are obtained from its role in the activity gathered from the depth videos and tracked by SIFT features. The results of this study assert that the anticipation, which is composed of the sub-activities and interactions with objects, enhances the object detection performance for past activities and affordances, nevertheless the performance declines drastically for future actions and affordances. Another empirical study investigating the effects of visual feedback in arm movements is performed in [169]. In the experiments, the human subjects move their arms between two goal stimuli with pointing-like dot-placing task at even spaces. The findings suggest that vision has effects on movement accuracy while former knowledge of target positions has smaller effect as well as the movement length has no influence on the performance.

The debate continues about the best strategies for the role of visual data in motor functions and the parameters that affect the visuomotor actions. However, the major part of the studies suggest that the vision and learned attributes have relation as well as the people make inferences combining them for the known and unknown situations.

Afterwards, the feedback is employed to adapt if there occurs any unforeseen failure. Remarkably, the algorithms are also disturbed by unknown situations. Therefore, our platform is supported by two visual feedbacks along with the encoders that the position and stiffness can be calculated. Moreover, the predefined attributes of objects make our system more stable and robust to disturbances. The assignment of the physical attributes and recognizing object classes contributes to the calculation of force vector to be applied to the objects identically. The force vector includes the position where the force will be applied on the object surface, the magnitude, and the orientation information.

4.3 The Robotic Mechanism and Simulations

Robots are employed for many decades in the industry; however, nowadays they occupy a significant space in everyday life. Both use cases expect more sensitivity, safety, robustness, and precision than the levels of these in the past. As a matter of fact the current requirements and necessities anticipate further meaningful semantic knowledge extraction from the data. The object class labels are deficient for robotics applications regarding the meaningful knowledge as long as they are not supported by different type of information such as the physical and semantic properties of objects. In other words, inserting additional information to the object labels facilitate the robots to approximate current expectations. Therefore, it is supposed from the robotics systems to be equipped with advanced hardware and software to execute vision and manipulation tasks. In this part of the thesis, the simulation results of a robotic arm, which manipulates the recognized objects using deep neural networks considering the physical features, are given for 10 different categories from the synthetic part of *ADORESet*. The robotic arm calculates the force to be applied onto the objects as given in Equation (4.1):

$$F(x,y,z) = c_1 * density + c_2 * flexibility + c_3 * deformability + c4 * friction * mass \tag{4.1}$$

where (x,y,z) is the position where the force will be applied, c_i for $i = 1,...,4$ denote the empirical coefficients that are adjusted according to the simulations identical to each object and other physical properties belong to the object categories as well. The miscalculation of this equation or using a fixed level of force value can cause three

possibilities concerning the results of the object manipulation tasks as follows: *i)* knocking over due to the excessive force, *ii)* remaining in the constant position and does not move due to the insufficient force, and *iii)* the satisfactory motion ending at the desired position because of the adequate force. The (x, y, z) coordinates are determined according to the object classes and the measured dimensions of the object from the cameras; thus, the center of mass position can be calculated approximately as the point where the force is applied. To conduct the simulation with the proposed manipulation approach, a 2 DoF robotic arm (one prismatic joint for linear vertical motion and one revolute joint for rotational motion) is employed by adopting the Gazebo simulation tool within robot operating system (ROS). The manipulation task is defined as to move the visually recognized objects from one point to another and the torque control of the manipulator is contemplated as on-off control. Generating the object specific behavior is accomplished by considering the additional object properties. A fine-tuned *VGGNet* model is used as the object recognition algorithm and a color-based object localization method is preferred to determine the spatial coordinates of the objects, which uses the depth data from three depth cameras located on top, front and one side of the *Deep Table* for position tracking and recognition, respectively. In Figure 4.2 a), the initial positions of these objects within the *Deep Table* is displayed and Figure 4.2 b) shows the view of the front camera and predicted labels of the objects remaining in front of them. After the manipulator performs the given tasks, the objects are displaced to the desired positions as illustrated in Figure 4.2 c). The displacement during the manipulation is given in Figure 4.2 d), which reveals how the algorithm generates a force application procedure according to the predicted labels and other properties belong to the objects. For instance, the movements of the smaller in volume and lighter in weight objects end before than the bigger in volume and heavier in weight objects. All the data is gathered in the computer which process and calculates four outputs; i) position, ii) stiffness of VRP-VSJ and position of the Cartesian platform in iii) x-axis and iv) y-axis. The serial port communication is adopted to transmit commands and receive sensory feedback concerning the simplicity.

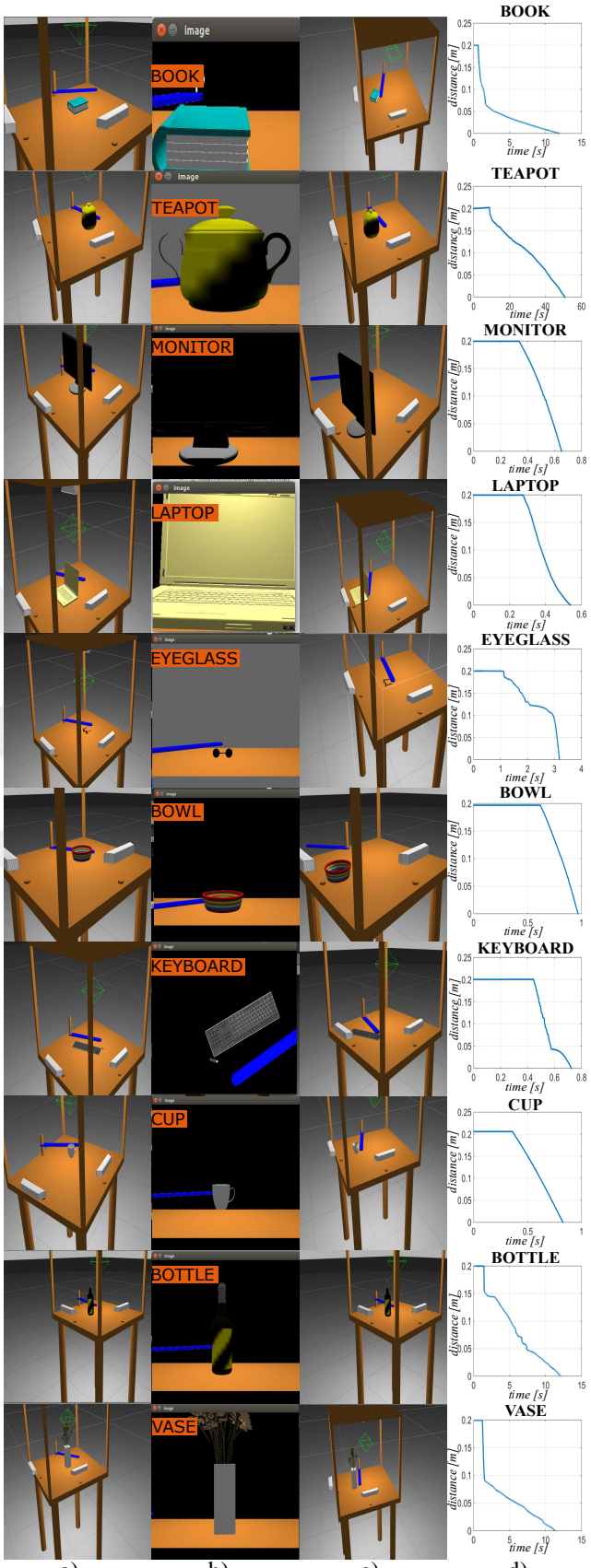


Figure 4.2 : The manipulation simulations within the *Deep Table*: a) initial conditions, b) object recognition, c) side view and object labels predicted by the algorithm, and d) the progress during the manipulation.

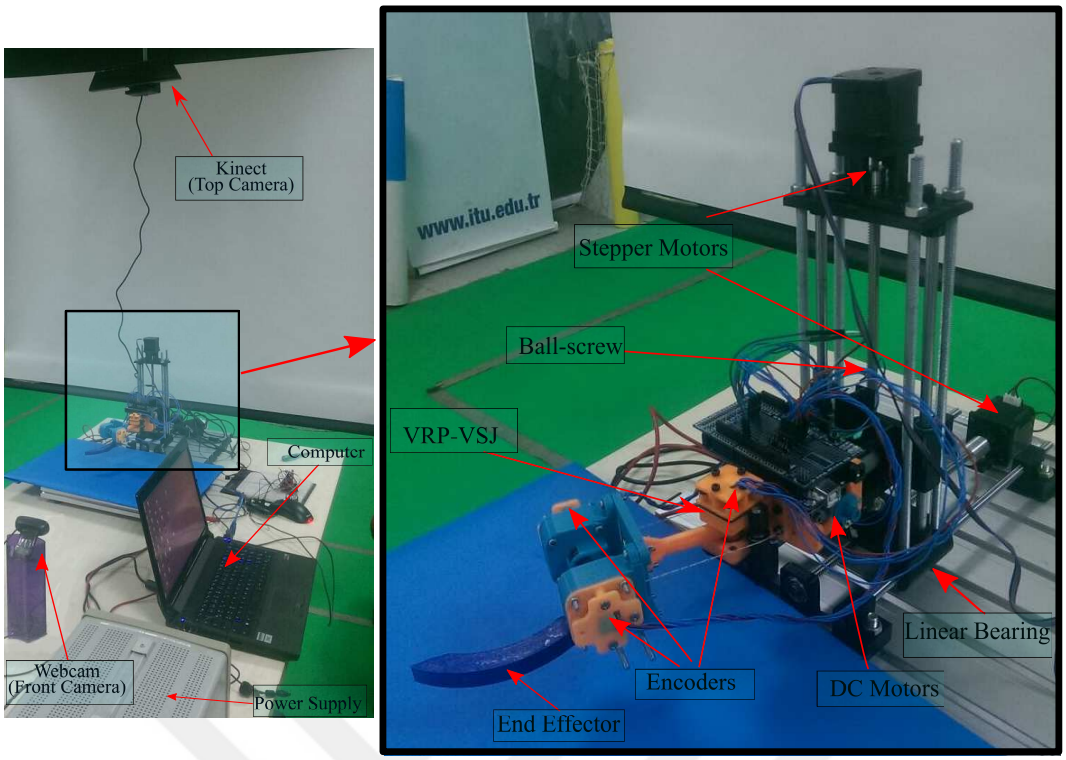


Figure 4.3 : The *Deep Table*.

4.4 The Robotic Mechanism, the *Deep Table* and Experiments

Since the main objective of this thesis is to manipulate the visually recognized objects, a 3 DoF robotic mechanism is constructed using 3D printed elements excluding off-the-shelf products. Thus, the cost of the complete system is minimized. The components of the *Deep Table* is presented in Figure 4.3.

There is a depth camera placed on top of the workspace at 100*cm* height focusing the object motion and the visual depth data is also used to calculate the volume and location of the objects. In addition, there is a webcam located in the horizontal direction of the end effector at 60*cm* distance and its data is used to recognize objects and localize the bounding box coordinates in the vertical plane, which is also used to determine the contact height. The computer gathers the data from the whole system and it is employed to monitor the progress during experiments. The base of the mechanism is fixed to the table and the horizontal axis is moved over a linear bearing. The vertical axis motion is maintained by driving a ball-screw. The motors, which control the vertical and horizontal axes, are nema-17 stepper-motors, which adjust the height and the horizontal depth of the mechanism. The VRP-VSJ is equipped with 3 encoders, the data of which is used to compute the applied f/t. There is an end

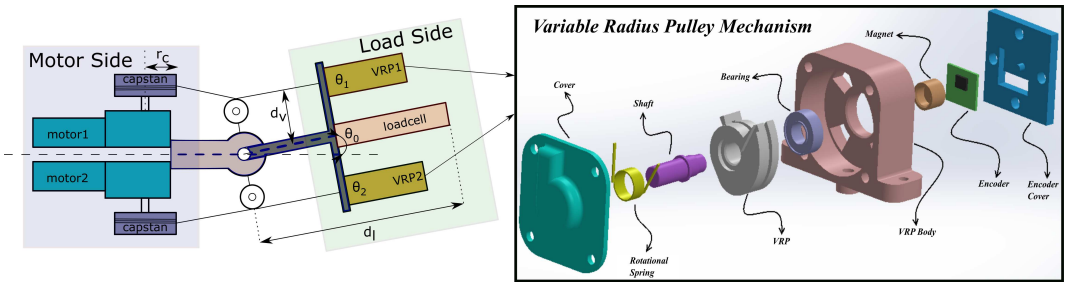


Figure 4.4 : The VRP-VSJ mechanism and a VRP in exploited view.

effector mounted to the VRP-VSJ to improve the contact to the objects. There are two DC-motors with worm-gears those control the stiffness and position of the VRP-VSJ mechanism via cables. The stepper-motors and DC-motors are controlled by Arduino boards. Given these points, the *Deep Table* provides an infrastructure for manipulation scenarios and testing computer vision algorithms involving robotic mechanisms. The VRP-VSJ contributes to the *Deep Table* by means of variable stiffness actuation ability, compliance and safety, energy-efficiency, simplicity, compactness and light-weight structure, minimum backlash, modularity, and human-like manipulation ability.

4.4.1 The VRP-VSJ mechanism and force estimation

VRP mechanisms maintain a broad spectrum of unique approaches to modern robotics research, mainly to the soft-robotics field. One of the most influential properties of VRP mechanisms is to have adjustable stiffness characteristics where springs are principally utilized to control the rigidity by a cable. Therefore, compliance and robustness can also be thought as the distinctive components of VRPs. Additionally, involving a compact design and being ready to be produced at a low cost contribute to design flexibility of VRPs. To benefit from the advantages of using a VRP-VSJ mechanism to manipulate the objects, we designed and manufactured a mechanism involving a VRP-VSJ tool as the manipulator part. The VRP-VSJ design used in this thesis is shown in Figure 4.4 with the exploited view of the VRP box that facilitates the adjust the stiffness.

The θ values show the angles of the joints, and VRPs. The stiffness adjustment is maintained by both the VRP profile and the rotational spring. Since the end effector of our mechanism does not equipped with a force measuring sensor, it cannot measure the f/t during the contact to the object and we have to estimate the applied f/t from the data of three encoders. Hence, we built a sparse ANN structure for estimation.

There are many successful model-based applications; however, uncertainties in system parameters and modeling assumptions cause model-based estimation methods to be error-prone. ANNs are adequate to produce outstanding results in the case of multi-channel and various kind of data in large quantities. Moreover, variance in the training data is a desired feature in most cases. However, noise is one of the most compelling issues for model-based architectures, ANNs have the competence to cope with noise in input data. The ANN model adopted in this study does external force estimation using only VRP-VSJ encoders data. Two ANN models according to the input data types are considered estimating the external force (simply a regression problem) effecting on the mechanism. One of the ANN models is fed with three encoders data belong to the joint and VRPs whilst the other one uses encoder data of the two motors additionally. Load cell measurements are assumed to be the ground-truth values for both models. We split the data into training and test parts according to k-fold cross validation convention where k is set to 5 as a typical usage. There are 17500 samples for each input feature within the whole dataset that is collected directly from the VRP-VSJ mechanism. Furthermore, our ANN models are trained on a laptop computer with a CPU of Intel i7 and a GPU Nvidia GTX-570m that runs Linux Ubuntu 16.04. We implemented the codes with Keras runing on top of Tensorflow. In Figure 4.5, the 5-input ANN model is symbolically represented. Correspondingly, the 3-input ANN model can be obtained by removing the last two input features. The joint angle is denoted by θ_0 while the VRP angles are indicated by θ_1 and θ_2. Additionally, the motor angles are represented by θ_{M1} and θ_{M2}.

The 3-input and 5-input models are trained for various number of hidden layers and neurons adopting different activation functions and backpropagation techniques. The best model, which has 3 inputs at 2 hidden layers having 80 neurons at each achieved a root-mean-square error (RMSE) of 0.0987 rate after 500 epochs of training. This model uses stochastic gradient descent for parameter update with 0.0005 learning rate and ReLU as activation function. The output of the network estimates the applied f/t values within a $[0, 1000]$ range in grams.

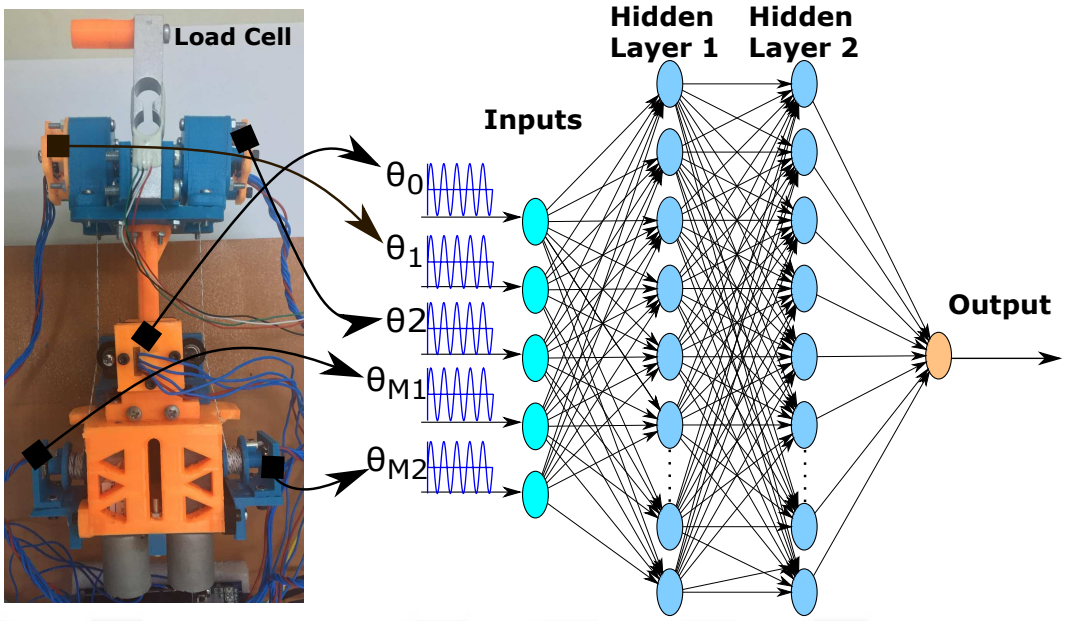


Figure 4.5 : The f/t estimation model training representation.

4.4.2 Proposed control methodology

The relationship between the appearance of objects and movement related anticipations for humans are widely investigated in the literature. But there is still not an exact consensus on how people use the visual data to perform actions towards objects and what affects the brain and visuomotor control functions. On the other hand, the robotics researchers study the similar subjects with the aim of bridging the gap between robot and human motions considering the qualitative and quantitative performance metrics such as smoothness, safety, efficiency, robustness, and accuracy. Hence, they investigate the role of visual feedback in human perception and action domains as well as a great amount of studies are carried on the human-inspired robotics field involving the human musculoskeletal system studies. Semantic knowledge about the target object for manipulation tasks provides a priori data before the contact or grasp; therefore, the action can be determined including the object dynamics. Since we intend to point out the benefits of this approach, a straightforward manipulation task is taken into consideration for the simulations and the experiments. An object which is placed on a table surface is requested to be moved by a precise value with the proposed variable-stiffness joint mechanism without tilting over or damaging the object. To complete the relocation successfully, approximate object weight and center of gravity are utilized to determine the joint stiffness and contact height. The suggested control block diagram is presented in Figure 4.6.

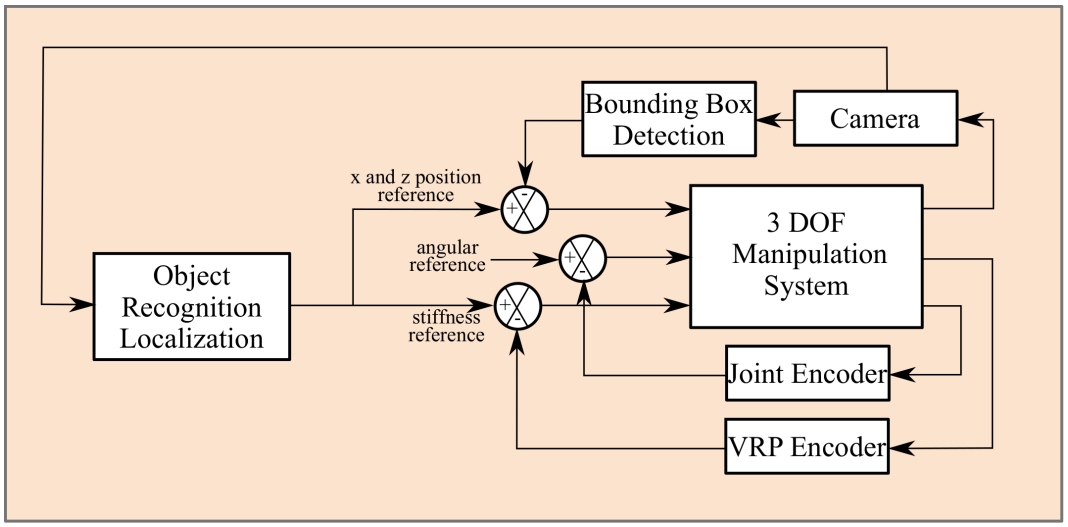


Figure 4.6 : The control schema of the proposed manipulation system.

Regarding the figure, after the object to be put on the Deep Table, first, it is recognized with our computer vision system containing object localization and recognition parts. A look-up table which keeps the physical information about the thirty objects included in ADORESet, is utilized to pick up the weight and the center of gravity of the object. Then required stiffness is calculated according to object weight. The computer vision algorithm does not only indicate the object type but also produces the bounding-box coordinates. As a result of processing the data from the depth and web cameras, the explicit location and the height of the object is detected, and the contact point is calculated, especially in vertical axis. The controllers of rigid Cartesian manipulator bring the end-effector to correct position while the VRP-VSJ carries out the moving task with specified stiffness. Considering the control architecture proposed herein and the human in manipulation task, it can be seen that there are similarities between them. As humans approach an identified object by taking into account the experiences, which can be accepted as the feedforward direction. Also, the shape of grasping or moving is notable for an outstanding operation; the experiences of the Newtonian dynamics determine it. On the other hand, after the contact, both the position and the f/t information is acquired by visual and tactile sensing units, and they are fed back to the main controller, brain, until the task is accomplished. Likewise, the recommended approach uses prior knowledge produced by the look-up table before starting the task. Then it provides back the visual and force data. If one of the feedforward or feedback paths are blocked, then the performance of the controller either drops immediately or fails. As it is expressed earlier, both motors in the VRP-VSJ mechanism adjusts

the position and stiffness independently. However, when its path is intercepted, it is impractical to manage the position and applied force in the same axis at the same time. Therefore, the coefficients of P controllers in both controller are regarded as the weighing factors. To accomplish the task, the influence of the stiffness control is intensified when compared to the position controller.

4.5 Results

The simulation approach lead us to gain some time before implementing our algorithms to manipulate visually recognized objects. A simplified version of our *Deep Table* is replicated in Gazebo environment. The findings suggests that assigning physical attributes to object identically improve the semantic information retrieval along with moving capability of the system. The comparison results of the situations are explained whether the robotic arm has the preliminary information about the category and physical attributes. The results reveal that satisfactory motions cannot be achieved except random situations if the preliminary information does not exist.

The empirical tests conducted in this section are presented for five of the many potential scenarios incorporating combinations of incorrect positioning and stiffness selection for high and low inertia objects. Firstly, the proposed control algorithm is disrupted deliberately to bring about inaccurate stiffness or position outputs to demonstrate and explore those probable situations. Then the algorithm is reset to defaults to indicate the performance of the proposed control technique and VRP-VSJ mechanism. The image sequences belong to the 5 experiments captured during the motion from both front camera and depth camera are shown in Figure 3.18. The individual images for each test starts with the initial representations of the components of the mechanism and the progress during the operation until the end is given with the images from top to the bottom, respectively. All of the test cases are expressed by 5 images. For each cases, the images at leftmost are captured from the front webcam and the remaining images are taken from the depth camera at the same time. While the middle images are ordinary color images, the rightmost images are the depth images. Fusing the information from these with the encoders, we perform the manipulation of visually recognized objects within the *Deep Table*. A cup is used in the first case as shown in Figure 4.7 a) and it is resulted in a failure in the end by tilting the cup over as can be

seen from the last two image sequences. On the other hand, Figure 4.7 b) displays a successful operation for cup moving task by adjusting the stiffness and the contact point properly. In Figure 4.7 c) and Figure 4.7 d), the empty bottles are employed to execute the manipulations. Due to the different f/t values applied onto the bottles from different contact points the test in Figure 4.7 c) accomplishes the tasks while the test in Figure 4.7 d) yields failure as tilting the bottle over. The last case expressed in this thesis is illustrated in Figure 4.7 e) where a semi-filled bottle is picked as the object. Unlike the other situations, the motion cannot be completed in behalf of insufficient f/t applied onto the object even if the contact position is correct and the object is not tilted over. If the bottle is removed from the path, then it completes its motion as simultaneously. Moreover, the resulting VRP-VSJ positions are also explained hereafter with the order of the cases given in Figure 4.7.

Figure 4.7 : The empirical tests conducted on the *Deep Table*: a) Sigmoid, b) Gradient of Sigmoid, c) Tanh , d) Gradient of Tanh, e) ReLU.

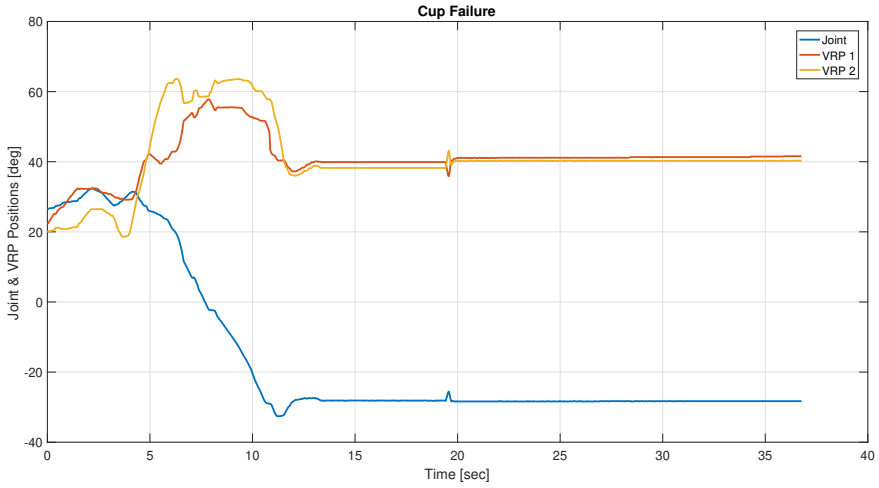


Figure 4.8 : The failed cup motion because of tilting over.

Even though it is an unequivocal task, the contact point between the robot and the object is crucial for the moving task. So, experience with Newton dynamics is indispensable to accomplish such a straightforward motion. While touching point under the center of gravity ensures the stability of the moving, higher points cause tilt over. The data gathered from the encoders of the VRP-VSJ mechanism during the experiment is presented in Figure 4.8 as a cup is knocked over. It is worth remarking that, the time of this figure starts just before the contact moment. The contact between the object and the arm is expressed in the difference of two VRP encoders. The differences between the VRPs are linked to the applied force linearly. Thus, it can be observed from the figure that the contact is lost before the end effector comes to the desired position. In this case, although stiffness selection is convenient for the object, herein a cup, the task fails because of the incorrect decision of the contact point.

Another experiment is engaged a cup as the object to be manipulated to illustrate the performance of the controller when the object has high inertia. Figure 4.9 displays the successful results of this test. Higher forces are applied to the object from the previous case as can be discovered from the graph. At the end of the operation, the arm arrives at the desired position, 30°.

In the preceding experiment, the default controller configuration is uploaded to test the successful moving task with the objects as discussed earlier. In this situation, when the inertia is small, a satisfactory level of stiffness applies an appropriate force to move the object carefully. Figure 4.10 demonstrates the successful experiment results of an

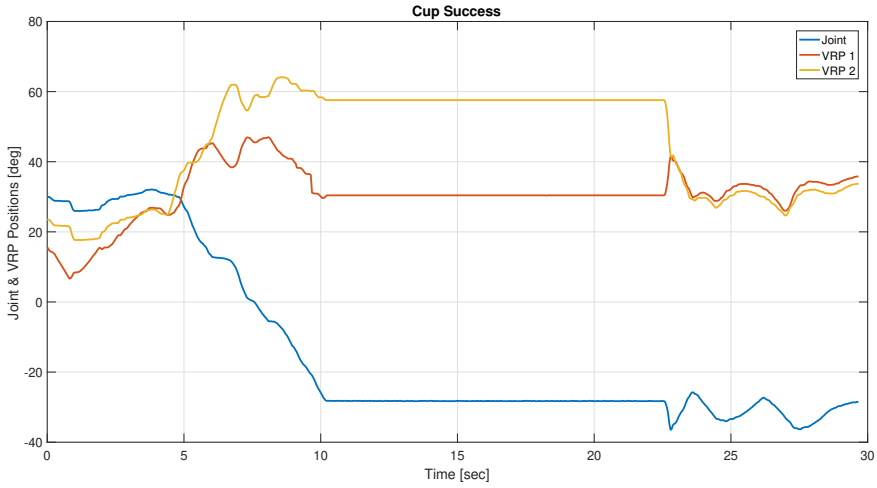


Figure 4.9 : The successful cup motion.

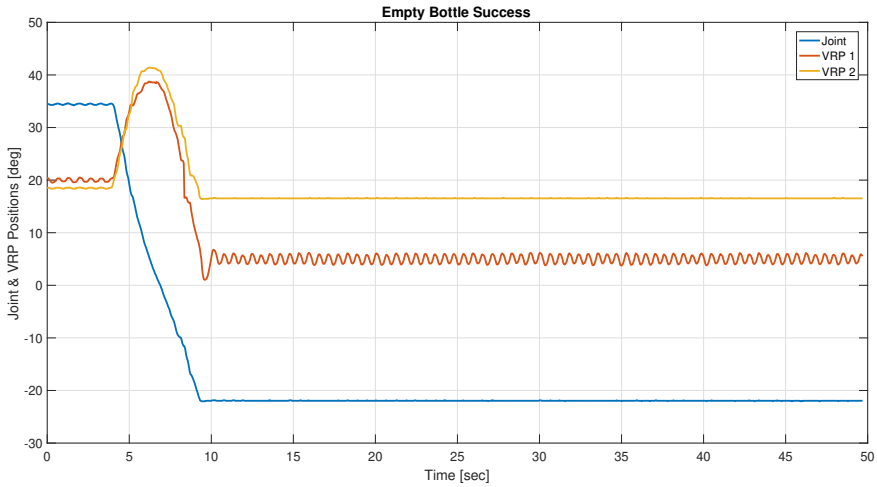


Figure 4.10 : The successful empty bottle motion.

empty bottle moving operation. It can be interpreted from the figure, during the contact phase which is between fifth and tenth seconds, the difference between both VPRs is small meaning that likewise the applied force is small. Since the inertia is expected to be the priority, the motion is not finished at the desired position; however, without occlusion of the object, it can not arrive at that position. Thus, it can move the object as close as possible to the desired location without damaging or tilting over it, which can be thought as a successful experiment.

This empirical test is to demonstrate how moving task is terminated when the stiffness is set to a larger value than needed amount for objects with smaller inertias. To interpret this, an empty bottle, which is weighing a few grams, is adopted as the object to be manipulated. The results of the same operation are expressed in Figure 4.11. As can be

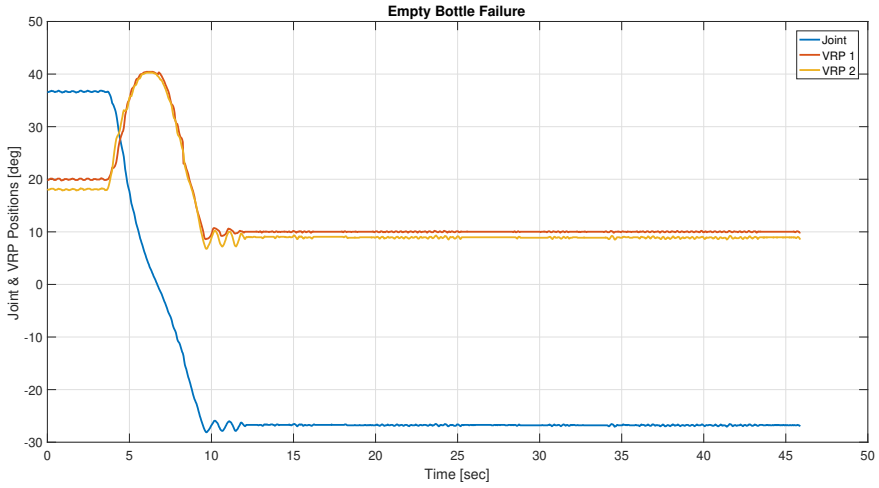


Figure 4.11 : The failed empty bottle motion because of tilting over.

seen that the impact appears at a moment around fifth second, where all of the encoder measurements intersect. At this situation, the applied force does not rise according to the encoders of VRPs. In fact, at the impact time, the applied force is big adequate to cause the contact to be unstable. Thereafter, the object tilts over, and VRP-VSJ maintains its motion without any further interaction.

As well as the greater stiffness values, the small stiffness also eventuates incomplete tasks. In this case of our empirical tests, the inertia of the bottle is increased by filling with water half of it. The results of the experiment are displayed in Figure 4.12. After the initial contact approximately at 25°, VRP-VSJ consumes a considerable effort to move the object to the desired position. However, it can be viewed that the motion of the arm ends at around -7°. In this test, the object is manually taken away to prove that the arm can pursue its movement when the massive bottle does not impede the way. Another aspect of this is the challenge of estimation of physical attributes of fillable objects such as bottle and cup despite the recognition of it. Even humans can fail in such a case, undoubtedly; they overcome this problem through feedback control.

4.6 Discussion and Conclusion

There is still not a solid unanimity on the link between the visual perception and visuomotor functions in humans. This particular area of the unique role of the object classes has been overlooked concerning the construction of a robot-object interaction framework. For this reason, we assert an alternative approach to the existing literature.

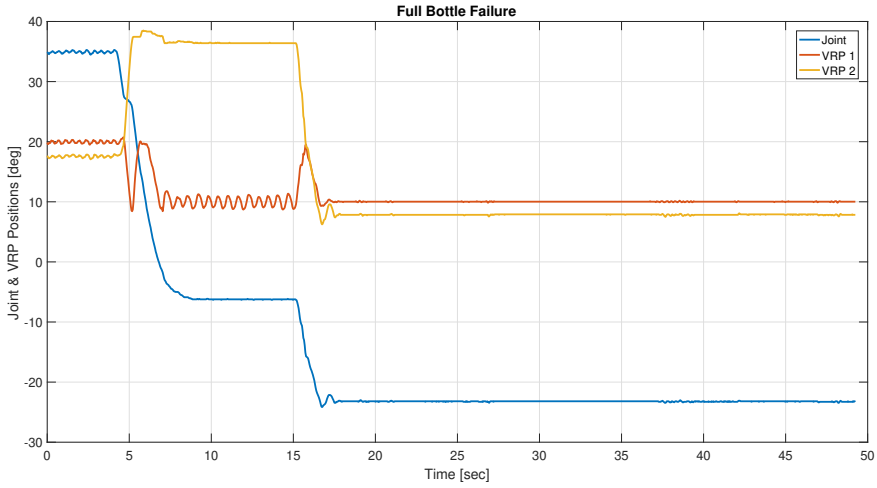


Figure 4.12 : The failed semi-filled bottle motion because of insufficient f/t.

To demonstrate the advantages of acquiring the semantic knowledge belonging to the objects identically, we propose to perform the suggested procedure in two stages as simulation and empirical tests, respectively.

The simulations are executed within a structured Gazebo environment equipped with a 2 DoF manipulator and three depth cameras as a *Deep Table* replication. The motion progresses for 10 object classes are presented comparatively. Once the objects are recognized using *VGG16* model, and then the manipulator acts according to the assigned features. Thus, the desired motions are achieved for all objects within different motion periods for the reason that the manipulator action characteristics differ for the other kind of objects. Unsuccessful manipulations are also observed due to the incorrect f/t applications and improper contact point determinations but not given amongst the results. The motion behaviors show characteristic properties according to the volume, weight, and contact points.

ADORESet provides a substantial amount of images per category and capability regarding the object types those can remain on a table surface. Employing the *ADORESet*, we trained two separate CNNs with the intention of object localization in spatial coordinates and object recognition. The fine-tuned architecture outputs show that the *RetinaNet* and *VGGNet* models are suitable for our experiments regarding the object localization and recognition, respectively. Thus, our system becomes able to process 4 frame-per-second and responds the bounding-box coordinates with the object label. Moreover, our *smart-pooling* algorithm has contributed the performance

as well. The 3 DoF robotic arm mechanism is capable of reaching every point within its workspace together with the stiffness adjustment ability. We demonstrated 5 separate experiments with the aim of clarification of our proposed approach. The generic structure of our procedure reveals how such a low-cost and modular robotic arm can manipulate different types of objects successfully adopting the visual feedback.

5. DISCUSSION AND CONCLUSION

Retrieving semantic information is necessary to implement complex robotic applications successfully. Thus, it is commonly expected from the robotics systems to be armed with advanced hardware and software. The modern humanoid robotics research exploits the nature and tries to mimic or replicate biological creatures. At this point, humans are the great observation sources according to this research field through fundamental properties to understand the working procedure of the brain and physical features to adapt different conditions. The interactions between these two properties are also a hot topic that still has blurred boundaries. It is commonly argued that the semantic information as a consequence of the physical interactions forms the perceptions and it manages the anticipations and future actions; however, the adaptation to new situations requiring different actions is argued as the opposite view that also affects the visuomotor functions. The visual perception is located at the heart of these discussions, which is accepted as one of the most influential data sources for the brain together with being the highest amount of data supplier. Therefore, we designed a system to implement object manipulation task adopting visual feedback and a VRP-VSJ mechanism those will facilitate our mechanism to approximate human manipulation approach.

As a starting point of examining the potential of the visual cues, first the conventional feature detector and descriptor combinations on image matching. Our universal structure for testing the performance outputs of feature detector and descriptors analyzes the combinations of these methods regarding the extra metrics such as minimum distance between proper matches, number of correct matches, orientation difference between matches. Then, it is showed that the response time of those algorithms are not convenient for real-time applications along with the deficient accuracy rates. However, these out-of-fashion algorithms are able to obtain semantic information and help the visual systems to be useful for robotic applications, whether they require a lot of computational power. To overcome these issues due to the nature

of conventional feature extraction techniques, we refer to employ deep neural networks that are able process larger quantities of data intent to run in real-time applications with high performance rates.

The prevalence of intelligent systems makes it feasible to acquire more voluminous data with many more dimensions than before. One of the reasons behind the success of deep neural networks is the capacity of processing a vast amount of data. In fact, the increase in the data amount also raises its success. Moreover, time spent during developments in robotics is a significant cost to be reduced, which restricts maneuverability and diversity. To optimize this period, the referred method is to take advantage of simulation environments, which reproduce real-world conditions as much as possible. In most cases, machine vision based problems in robotics such as object detection and recognition, object tracking and manipulation are implemented employing real-world or simulation images, separately. The primary purpose of computer vision and control in the robotics field is to obtain a perception and cognition proficiency comparable to or better than humans. In this sense, incorporation of additional object classes in image datasets, for example, wild animals, large structures, big vehicles, etc. is futile which lead to worse performance results. Although the robotic applications simulations give successful results, the outcomes cannot be instantaneously applied in real-world tests or end-user products due to the inconsistencies between real and simulation environments. For this reason, we proposed *ADORESet* that is composed of colored images, which has 30 classes with the dimension of 300×300 pixels. Each class has 2500 real-world images acquired from wild web and 750 synthetic images that are generated within Gazebo. This hybrid dataset enables researchers to implement their algorithms both for real-world and simulation environment conditions. Our hybrid dataset is fully-annotated, and the limits of objects are manually specified, and bounding box coordinates are provided. Successor objects are also labeled to give statistical information about the relations of the objects within the dataset. For example, the relationship between monitor, keyboard, and mouse can be directly obtained using this useful information. *ADORESet* should be of interest to the field of robotics researchers by means of its hybrid form, compactness in terms of lightweight and relevancy to further robotics and computer vision applications.

Even though CNN architectures have many advantages, training and test steps require a lot of computational effort. Using pretrained parameters and/or ensembles of different models are more preferred rather than the end-to-end training of CNNs from scratch. Most of the improvements have benefited from previous architectures and then fine-tuned or converted them to a more reasonable form for specific purposes. Hence, we exploited the advantage of having ADORESet and test the state-of-the-art CNNs for object recognition regarding the dataset constitution whether it is composed of synthetic, real or hybrid images. We also declared a pooling method called *smart-pooling* that extends the current literature with its transitive structure. Furthermore, the pooling layer also contribute the regularization performance of larger CNN architectures that also helps to prevent over-fitting. We re-trained 36 CNNs in total and the *VGGNet* model is adopted for object recognition in our tests depending on the performance. The results show that the data constitution is vital for the object recognition accuracy together with the quantity of images per category. But the ADORESet enabled us to train the CNN models once, and then, utilize the parameters either for simulation or real-world applications without any restrictions depending on the data type. Moreover, with the intention of predicting the bounding-box spatial coordinates surrounding the objects, we fine-tuned the *RetinaNet* architecture to localize the objects in the images for specifying the place of the objects in the manipulation scenarios. In addition to the visual cues, we assigned extra features to objects considering their physical characteristics. Our vision system hereby has ability to connect the labels and locations of the recognized objects with the assigned physical attributes. So that our robotic mechanism inside the *Deep Table* can move different types of objects from one point to another by itself with one algorithm.

The *Deep Table* is a special test platform that is rigged with a depth camera on the top, a webcam in the front, a 3 DoF robotic mechanism for manipulation involving the VRP-VSJ system, motors and controller boards together with the computer. In manipulation operation, the contact points to the objects play an important role for successful motions in the context of satisfactory movement, time and energy consumption. If the optimum point for the force application can be determined, then the robot will consume less time and less energy to move the object from its path. Exploiting the knowledge extracted from the sensors within the Deep Table our control

algorithm executes the manipulation. We present 5 of the many possible cases for the empirical tests that prove the plausibility and performance of our algorithm. The results indicate that our algorithm can move different types of objects successfully ranging from several grams (empty bottle) to around 250 grams (ceramic cup). The experiments also show the role of contact point where the f/t is applied onto the object. If the contact point is adjusted conveniently, then the manipulation is terminated with a tilt over of the object.

In essence, in this thesis, a system is proposed for the manipulation of visually recognized objects using deep neural networks. In addition to contributing to the current literature by the comprehensive comparison of feature extraction method combinations, the ADORESet, the smart-pooling, and the control approach, we open a path to the potential studies such as developing further object detection and recognition algorithms, making the detailed performance analysis of the *smart-pooling*, improving the control techniques involving reinforcement learning for object manipulation within the *Deep Table*, upgrading the VRP-VSJ mechanism to a higher DoF.

REFERENCES

[1] **Dahl, A.L.**, **Aanæs, H. and Pedersen, K.S.** (2011). Finding the best feature detector-descriptor combination, *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2011 International Conference on*, IEEE, pp.318–325.

[2] **Url-1**, `http://www.mcescher.com/gallery/back-in-holland/relativity/`, access Date: 31.01.2018.

[3] **Shepard, R.N.** (1978). The mental image., *American Psychologist*, *33*(2), 125.

[4] **Pylyshyn, Z.W.** (1973). What the mind's eye tells the mind's brain: A critique of mental imagery., *Psychological Bulletin*, *80*(1), 1.

[5] **Gregory, R.L.** (2015). *Eye and brain: The psychology of seeing*, Princeton University Press.

[6] **Jolicoeur, P.**, **Gluck, M.A. and Kosslyn, S.M.** (1984). Pictures and names: Making the connection, *Cognitive Psychology*, *16*(2), 243–275.

[7] **Marszalek, M. and Schmid, C.** (2007). Semantic hierarchies for visual object recognition, *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, IEEE, pp.1–7.

[8] **Cheong, H.**, **Park, S.**, **Park, M. and Park, S.K.** (2007). Vision-based global localization in indoor environment with an object entity-based hybrid map, *Control, Automation and Systems, 2007. ICCAS'07. International Conference on*, IEEE, pp.218–223.

[9] **Viswanathan, P.**, **Meger, D.**, **Southey, T.**, **Little, J.J. and Mackworth, A.K.** (2009). Automated spatial-semantic modeling with applications to place labeling and informed search, *Computer and Robot Vision, 2009. CRV'09. Canadian Conference on*, IEEE, pp.284–291.

[10] **Tenorth, M.**, **Kunze, L.**, **Jain, D. and Beetz, M.** (2010). Knowrob-map-knowledge-linked semantic object maps, *Humanoid Robots (Humanoids), 2010 10th IEEE-RAS International Conference on*, IEEE, pp.430–435.

[11] **Rogers, J.G.**, **Trevor, A.J.**, **Nieto-Granda, C. and Christensen, H.I.** (2011). Simultaneous localization and mapping with learned object recognition and semantic data association, *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, IEEE, pp.1264–1270.

[12] **Filliat, D.**, **Battesti, E.**, **Bazeille, S.**, **Duceux, G.**, **Gepperth, A.**, **Harrath, L.**, **Jebari, I.**, **Pereira, R.**, **Tapus, A.**, **Meyer, C.** *et al.* (2012). RGBD object recognition and visual texture classification for indoor semantic mapping, *Technologies for Practical Robot Applications (TePRA), 2012 IEEE International Conference on*, IEEE, pp.127–132.

[13] **Pangercic, D.**, **Pitzer, B.**, **Tenorth, M. and Beetz, M.** (2012). Semantic object maps for robotic housework-representation, acquisition and use, *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, pp.4644–4651.

[14] **Stückler, J.**, **Biresev, N. and Behnke, S.** (2012). Semantic mapping using object-class segmentation of RGB-D images, *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, IEEE, pp.3005–3010.

[15] **Su, Y.**, **Allan, M. and Jurie, F.** (2010). Improving object classification using semantic attributes., *BMVC*, pp.1–10.

[16] **Li, L.J.**, **Su, H.**, **Lim, Y. and Fei-Fei, L.** (2014). Object bank: An object-level image representation for high-level visual recognition, *International Journal of Computer Vision*, *107*(1), 20–39.

[17] **Deng, J.**, **Berg, A.C. and Fei-Fei, L.** (2011). Hierarchical semantic indexing for large scale image retrieval, *Computer Vision and Pattern Recognition (CVPR), 2011 IEEE Conference on*, IEEE, pp.785–792.

[18] **Liu, J.** (2013). Image retrieval based on bag-of-words model, *arXiv preprint arXiv:1304.5168*.

[19] **Chechik, G.**, **Sharma, V.**, **Shalit, U. and Bengio, S.** (2010). Large scale online learning of image similarity through ranking, *Journal of Machine Learning Research*, *11*(Mar), 1109–1135.

[20] **Liu, Y.**, **Zhang, D. and Lu, G.** (2008). Region-based image retrieval with high-level semantics using decision tree learning, *Pattern Recognition*, *41*(8), 2554–2570.

[21] **Shotton, J.**, **Johnson, M. and Cipolla, R.** (2008). Semantic texton forests for image categorization and segmentation, *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, IEEE, pp.1–8.

[22] **Choudhary, S.**, **Trevor, A.J.**, **Christensen, H.I. and Dellaert, F.** (2014). SLAM with object discovery, modeling and mapping, *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, IEEE, pp.1018–1025.

[23] **Herbst, E.**, **Henry, P.**, **Ren, X. and Fox, D.** (2011). Toward object discovery and modeling via 3-d scene comparison, *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, pp.2623–2629.

[24] **Stückler, J. and Behnke, S.** (2013). Hierarchical Object Discovery and Dense Modelling From Motion Cues in RGB-D Video, *IJCAI*, pp.2502–2509.

[25] **Sturm, J.** (2013). *Approaches to Probabilistic Model Learning for Mobile Manipulation Robots*, volume 89, Springer.

[26] **Jiang, Y.**, **Moseson, S. and Saxena, A.** (2011). Efficient grasping from rgbd images: Learning using a new rectangle representation, *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, IEEE, pp.3304–3311.

[27] **Taylor, G. and Kleeman, L.** (2008). *Visual perception and robotic manipulation: 3D object recognition, tracking and hand-eye coordination*, volume 26, Springer.

[28] **Ramanathan, V.**, **Li, C.**, **Deng, J.**, **Han, W.**, **Li, Z.**, **Gu, K.**, **Song, Y.**, **Bengio, S.**, **Rosenberg, C. and Fei-Fei, L.** (2015). Learning semantic relationships for better action retrieval in images, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.1100–1109.

[29] **Schmitz, A.**, **Bansho, Y.**, **Noda, K.**, **Iwata, H.**, **Ogata, T. and Sugano, S.** (2014). Tactile object recognition using deep learning and dropout, *Humanoid Robots (Humanoids), 2014 14th IEEE-RAS International Conference on*, IEEE, pp.1044–1050.

[30] **McCormac, J.**, **Handa, A.**, **Davison, A. and Leutenegger, S.** (2017). SemanticFusion: Dense 3D semantic mapping with convolutional neural networks, *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, pp.4628–4635.

[31] **Liao, Y.**, **Kodagoda, S.**, **Wang, Y.**, **Shi, L. and Liu, Y.** (2016). Understand scene categories by objects: A semantic regularized scene classifier using convolutional neural networks, *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, IEEE, pp.2318–2325.

[32] **Furuta, Y.**, **Wada, K.**, **Murooka, M.**, **Nozawa, S.**, **Kakiuchi, Y.**, **Okada, K. and Inaba, M.** (2016). Transformable semantic map based navigation using autonomous deep learning object segmentation, *Humanoid Robots (Humanoids), 2016 IEEE-RAS 16th International Conference on*, IEEE, pp.614–620.

[33] **Himstedt, M. and Maehle, E.** (2017). Online semantic mapping of logistic environments using RGB-D cameras, *International Journal of Advanced Robotic Systems*, *14*(4), 1729881417720781.

[34] **Ma, L.**, **Stückler, J.**, **Kerl, C. and Cremers, D.** (2017). Multi-view deep learning for consistent semantic mapping with rgb-d cameras, *ArXiv Preprint arXiv:1703.08866*.

[35] **Lowe, D.G.** (2004). Distinctive image features from scale-invariant keypoints, *International Journal of Computer Vision*, *60*(2), 91–110.

[36] **Bay, H.**, **Ess, A.**, **Tuytelaars, T. and Van Gool, L.** (2008). Speeded-up robust features (SURF), *Computer Vision and Image Understanding*, *110*(3), 346–359.

[37] **Fei-Fei, L. and Perona, P.** (2005). A bayesian hierarchical model for learning natural scene categories, *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 2, IEEE, pp.524–531.

[38] **Harris, C. and Stephens, M.** (1988). A combined corner and edge detector., *Alvey Vision Conference*, volume 15, Citeseer, pp.10–5244.

[39] **Canny, J.**, (1987). A computational approach to edge detection, Readings in Computer Vision, Elsevier, pp.184–203.

[40] **Leutenegger, S.**, **Chli, M. and Siegwart, R.Y.** (2011). BRISK: Binary robust invariant scalable keypoints, *Computer Vision (ICCV), 2011 IEEE International Conference on*, IEEE, pp.2548–2555.

[41] **Rosten, E. and Drummond, T.** (2005). Fusing points and lines for high performance tracking, *Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on*, volume 2, IEEE, pp.1508–1515.

[42] **Rosten, E. and Drummond, T.** (2006). Machine learning for high-speed corner detection, *European Conference on Computer Vision*, Springer, pp.430–443.

[43] **Rublee, E.**, **Rabaud, V.**, **Konolige, K. and Bradski, G.** (2011). ORB: An efficient alternative to SIFT or SURF, *Computer Vision (ICCV), 2011 IEEE international conference on*, IEEE, pp.2564–2571.

[44] **Calonder, M.**, **Lepetit, V.**, **Strecha, C. and Fua, P.** (2010). Brief: Binary robust independent elementary features, *European Conference on Computer Vision*, Springer, pp.778–792.

[45] **Dalal, N. and Triggs, B.** (2005). Histograms of oriented gradients for human detection, *Computer Vision and Pattern Recognition, 2005. CVPR 2005. IEEE Computer Society Conference on*, volume 1, IEEE, pp.886–893.

[46] **BAYRAKTAR, E. and BOYRAZ, P.** (2017). Analysis of feature detector and descriptor combinations with a localization experiment for various performance metrics, *Turkish Journal of Electrical Engineering & Computer Sciences*, *25*(3), 2444–2454.

[47] **Mikolajczyk, K. and Schmid, C.** (2005). A performance evaluation of local descriptors, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *27*(10), 1615–1630.

[48] **Mikolajczyk, K.**, **Tuytelaars, T.**, **Schmid, C.**, **Zisserman, A.**, **Matas, J.**, **Schaffalitzky, F.**, **Kadir, T. and Van Gool, L.** (2005). A comparison of affine region detectors, *International Journal of Computer Vision*, *65*(1-2), 43–72.

[49] **Remondino, F.** (2006). Detectors and descriptors for photogrammetric applications, *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, *36*(3), 49–54.

[50] **Figat, J.**, **Kornuta, T. and Kasprzak, W.** (2014). Performance evaluation of binary descriptors of local features, *International Conference on Computer Vision and Graphics*, Springer, pp.187–194.

[51] **Mikolajczyk, K. and Schmid, C.** (2004). Comparison of affine-invariant local detectors and descriptors, *Signal Processing Conference, 2004 12th European*, IEEE, pp.1729–1732.

[52] **Iscen, A.**, **Tolias, G.**, **Gosselin, P.H. and Jégou, H.** (2015). A comparison of dense region detectors for image search and fine-grained classification, *IEEE Transactions on Image Processing*, *24*(8), 2369–2381.

[53] **Guclu, O. and Can, A.B.** (2015). A comparison of feature detectors and descriptors in rgb-d slam methods, *International Conference Image Analysis and Recognition*, Springer, pp.297–305.

[54] **Hartmann, J.**, **Klussendorff, J.H. and Maehle, E.** (2013). A comparison of feature descriptors for visual SLAM, *Mobile Robots (ECMR), 2013 European Conference on*, IEEE, pp.56–61.

[55] **Huang, A.S.**, **Bachrach, A.**, **Henry, P.**, **Krainin, M.**, **Maturana, D.**, **Fox, D. and Roy, N.**, (2017). Visual odometry and mapping for autonomous flight using an RGB-D camera, Robotics Research, Springer, pp.235–252.

[56] **Wang, X.**, **Vozar, S. and Olson, E.** (2017). FLAG: Feature-based Localization between Air and Ground, *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, pp.3178–3184.

[57] **Krajník, T.**, **Cristóforis, P.**, **Kusumam, K.**, **Neubert, P. and Duckett, T.** (2017). Image features for visual teach-and-repeat navigation in changing environments, *Robotics and Autonomous Systems*, *88*, 127–141.

[58] **Schmidt, A. and Kraft, M.**, (2015). The impact of the image feature detector and descriptor choice on visual slam accuracy, Image Processing & Communications Challenges 6, Springer, pp.203–210.

[59] **Türkmen, H.**, **Öksüzömer, N.**, **Yigit, C.B.**, **Bayraktar, E.**, **Küçük, H. and Boyraz, P.** (2016). Robust and smooth colour tracking by a tendon driven humanoid neck, *8th International Conference on Image Processing, Wavelet and Applications*.

[60] **Boyraz, P.**, **Yiğit, C.B. and Biçer, H.O.** (2013). UMAY 1: A modular humanoid platform for education and rehabilitation of children with autism spectrum disorders, *Control Conference (ASCC), 2013 9th Asian*, IEEE, pp.1–6.

[61] **Bartels, R.H.**, **Beatty, J.C. and Barsky, B.A.** (1998). Hermite and cubic spline interpolation, *An Introduction to Splines for Use in Computer Graphics and Geometric Modelling*, 9–17.

[62] **Matas, J.**, **Chum, O.**, **Urban, M. and Pajdla, T.** (2004). Robust wide-baseline stereo from maximally stable extremal regions, *Image and Vision Computing*, *22*(10), 761–767.

[63] **Tola, E.**, **Lepetit, V. and Fua, P.** (2008). A fast local descriptor for dense matching, *Computer Vision and Pattern Recognition, 2008. CVPR 2008. IEEE Conference on*, IEEE, pp.1–8.

[64] **Tola, E.**, **Lepetit, V. and Fua, P.** (2010). Daisy: An efficient dense descriptor applied to wide-baseline stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *32*(5), 815–830.

[65] **Winder, S.**, **Hua, G. and Brown, M.** (2009). Picking the best daisy, *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*, IEEE, pp.178–185.

[66] **Winder, S.A. and Brown, M.** (2007). Learning local image descriptors, *Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on*, IEEE, pp.1–8.

[67] **Alahi, A.**, **Ortiz, R. and Vandergheynst, P.** (2012). Freak: Fast retina keypoint, *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE conference on*, Ieee, pp.510–517.

[68] **Agrawal, M.**, **Konolige, K. and Blas, M.R.** (2008). Censure: Center surround extremas for realtime feature detection and matching, *European Conference on Computer Vision*, Springer, pp.102–115.

[69] **Mitchell, T.M.** (2016). *Machine Learning*, MIT Press, Cambridge MA, 1. edition.

[70] **Goodfellow, I.**, **Bengio, Y.**, **Courville, A. and Bengio, Y.** (2016). *Deep learning*, volume 1, MIT Press Cambridge.

[71] **Srivastava, N.**, **Hinton, G.**, **Krizhevsky, A.**, **Sutskever, I. and Salakhutdinov, R.** (2014). Dropout: A simple way to prevent neural networks from overfitting, *The Journal of Machine Learning Research*, *15*(1), 1929–1958.

[72] **LeCun, Y.**, **Bengio, Y. and Hinton, G.** (2015). Deep learning, *nature*, *521*(7553), 436.

[73] **LeCun, Y.**, **Bottou, L.**, **Bengio, Y. and Haffner, P.** (1998). Gradient-based learning applied to document recognition, *Proceedings of the IEEE*, *86*(11), 2278–2324.

[74] **LeCun, Y.** (1998). The MNIST database of handwritten digits, *http://yann. lecun. com/exdb/mnist/*.

[75] **Russakovsky, O.**, **Deng, J.**, **Su, H.**, **Krause, J.**, **Satheesh, S.**, **Ma, S.**, **Huang, Z.**, **Karpathy, A.**, **Khosla, A.**, **Bernstein, M.** *et al.* (2015). Imagenet large scale visual recognition challenge, *International Journal of Computer Vision*, *115*(3), 211–252.

[76] **Krizhevsky, A.**, **Sutskever, I. and Hinton, G.E.** (2012). Imagenet classification with deep convolutional neural networks, *Advances in Neural Information Processing Systems*, pp.1097–1105.

[77] **Zeiler, M.D. and Fergus, R.** (2014). Visualizing and understanding convolutional networks, *European Conference on Computer Vision*, Springer, pp.818–833.

[78] **Simonyan, K. and Zisserman, A.** (2014). Very deep convolutional networks for large-scale image recognition, *ArXiv Preprint ArXiv:1409.1556*.

[79] **Szegedy, C.**, **Liu, W.**, **Jia, Y.**, **Sermanet, P.**, **Reed, S.**, **Anguelov, D.**, **Erhan, D.**, **Vanhoucke, V.**, **Rabinovich, A.** *et al.* (2015). Going deeper with convolutions, CVPR.

[80] **Szegedy, C.**, **Vanhoucke, V.**, **Ioffe, S.**, **Shlens, J. and Wojna, Z.** (2016). Rethinking the inception architecture for computer vision, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2818–2826.

[81] **He, K.**, **Zhang, X.**, **Ren, S. and Sun, J.** (2016). Deep residual learning for image recognition, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.770–778.

[82] **Chollet, F.** (2016). Xception: Deep learning with depthwise separable convolutions, *ArXiv Preprint*.

[83] **Szegedy, C.**, **Ioffe, S.**, **Vanhoucke, V. and Alemi, A.A.** (2017). Inception-v4, inception-resnet and the impact of residual connections on learning., *AAAI*, volume 4, p. 12.

[84] **Xie, S.**, **Girshick, R.**, **Dollár, P.**, **Tu, Z. and He, K.** (2017). Aggregated residual transformations for deep neural networks, *Computer Vision and Pattern Recognition (CVPR), 2017 IEEE Conference on*, IEEE, pp.5987–5995.

[85] **Szegedy, C.**, **Toshev, A. and Erhan, D.** (2013). Deep neural networks for object detection, *Advances in Neural Information Processing Systems*, Curran Associates, Inc., pp.2553–2561.

[86] **Girshick, R.**, **Donahue, J.**, **Darrell, T. and Malik, J.** (2014). Rich feature hierarchies for accurate object detection and semantic segmentation, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.580–587.

[87] **Girshick, R.** (2015). Fast R-CNN, *Computer Vision (ICCV), 2015 IEEE International Conference on*, IEEE, pp.1440–1448.

[88] **Uijlings, J.R.**, **Van De Sande, K.E.**, **Gevers, T. and Smeulders, A.W.** (2013). Selective search for object recognition, *International Journal of Computer Vision*, *104*(2), 154–171.

[89] **Ren, S.**, **He, K.**, **Girshick, R. and Sun, J.** (2015). Faster r-cnn: Towards real-time object detection with region proposal networks, *Advances in Neural Information Processing Systems*, pp.91–99.

[90] **Redmon, J.**, **Divvala, S.**, **Girshick, R. and Farhadi, A.** (2016). You only look once: Unified, real-time object detection, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.779–788.

[91] **Liu, W.**, **Anguelov, D.**, **Erhan, D.**, **Szegedy, C.**, **Reed, S.**, **Fu, C.Y. and Berg, A.C.** (2016). Ssd: Single shot multibox detector, *European Conference on Computer Vision*, Springer, pp.21–37.

[92] **Lin, T.Y.**, **Dollár, P.**, **Girshick, R.**, **He, K.**, **Hariharan, B. and Belongie, S.** (2017). Feature pyramid networks for object detection, *Conference on Computer Vision and Pattern Recognition*, pp.2117–2125.

[93] **Lin, T.Y.**, **Goyal, P.**, **Girshick, R.**, **He, K. and Dollár, P.** (2017). Focal loss for dense object detection, *ArXiv Preprint ArXiv:1708.02002*.

[94] **Viola, P. and Jones, M.** (2001). Rapid object detection using a boosted cascade of simple features, *Computer Vision and Pattern Recognition, 2001. CVPR 2001. Proceedings of the 2001 IEEE Computer Society Conference on*, volume 1, IEEE, pp.I–I.

[95] **Lin, M.**, **Chen, Q. and Yan, S.** (2013). Network in network, *arXiv preprint arXiv:1312.4400*.

[96] **Sermanet, P.**, **Eigen, D.**, **Zhang, X.**, **Mathieu, M.**, **Fergus, R. and LeCun, Y.** (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks, *ArXiv Preprint ArXiv:1312.6229*.

[97] **He, K.**, **Zhang, X.**, **Ren, S. and Sun, J.** (2014). Spatial pyramid pooling in deep convolutional networks for visual recognition, *European Conference on Computer Vision*, Springer, pp.346–361.

[98] **Dai, J.**, **Li, Y.**, **He, K. and Sun, J.** (2016). R-fcn: Object detection via region-based fully convolutional networks, *Advances in Neural Information Processing Systems*, pp.379–387.

[99] **Iandola, F.**, **Moskewicz, M.**, **Karayev, S.**, **Girshick, R.**, **Darrell, T. and Keutzer, K.** (2014). Densenet: Implementing efficient convnet descriptor pyramids, *ArXiv Preprint ArXiv:1404.1869*.

[100] **Iandola, F.N.**, **Han, S.**, **Moskewicz, M.W.**, **Ashraf, K.**, **Dally, W.J. and Keutzer, K.** (2016). SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and< 0.5 MB model size, *ArXiv Preprint ArXiv:1602.07360*.

[101] **Huang, G.**, **Sun, Y.**, **Liu, Z.**, **Sedra, D. and Weinberger, K.Q.** (2016). Deep networks with stochastic depth, *European Conference on Computer Vision*, Springer, pp.646–661.

[102] **Larsson, G.**, **Maire, M. and Shakhnarovich, G.** (2016). Fractalnet: Ultra-deep neural networks without residuals, *ArXiv Preprint ArXiv:1605.07648*.

[103] **He, K.**, **Gkioxari, G.**, **Dollár, P. and Girshick, R.** (2017). Mask r-cnn, *Computer Vision (ICCV), 2017 IEEE International Conference on*, IEEE, pp.2980–2988.

[104] **Howard, A.G.**, **Zhu, M.**, **Chen, B.**, **Kalenichenko, D.**, **Wang, W.**, **Weyand, T.**, **Andreetto, M. and Adam, H.** (2017). Mobilenets: Efficient convolutional neural networks for mobile vision applications, *ArXiv Preprint ArXiv:1704.04861*.

[105] **Miller, G.A.** (1995). WordNet: a lexical database for English, *Communications of the ACM*, *38*(11), 39–41.

[106] **Lin, T.Y.**, **Maire, M.**, **Belongie, S.**, **Hays, J.**, **Perona, P.**, **Ramanan, D.**, **Dollár, P. and Zitnick, C.L.** (2014). Microsoft coco: Common objects in context, *European Conference on Computer Vision*, Springer, pp.740–755.

[107] **Everingham, M.**, **Eslami, S.M.A.**, **Van Gool, L.**, **Williams, C.K.I.**, **Winn, J. and Zisserman, A.** (2015). The Pascal Visual Object Classes Challenge: A Retrospective, *International Journal of Computer Vision*, *111*(1), 98–136.

[108] **Fei-Fei, L.**, **Fergus, R. and Perona, P.** (2006). One-shot learning of object categories, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *28*(4), 594–611.

[109] **Griffin, G.**, **Holub, A. and Perona, P.** (2007). *Caltech-256 object category dataset*.

[110] **Krizhevsky, A. and Hinton, G.** (2009). *Learning multiple layers of features from tiny images*.

[111] **Torralba, A.**, **Fergus, R. and Freeman, W.T.** (2008). 80 million tiny images: A large data set for nonparametric object and scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *30*(11), 1958–1970.

[112] **Calli, B.**, **Singh, A.**, **Walsman, A.**, **Srinivasa, S.**, **Abbeel, P. and Dollar, A.M.** (2015). The ycb object and model set: Towards common benchmarks for manipulation research, *Advanced Robotics (ICAR), 2015 International Conference on*, IEEE, pp.510–517.

[113] **Johnson-Roberson, M.**, **Barto, C.**, **Mehta, R.**, **Sridhar, S.N.**, **Rosaen, K. and Vasudevan, R.** (2017). Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks?, *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, pp.746–753.

[114] **Handa, A.**, **Patraucean, V.**, **Badrinarayanan, V.**, **Stent, S. and Cipolla, R.** (2015). SceneNet: Understanding real world indoor scenes with synthetic data, *ArXiv Preprint ArXiv:1511.07041*.

[115] **Netzer, Y.**, **Wang, T.**, **Coates, A.**, **Bissacco, A.**, **Wu, B. and Ng, A.Y.** (2011). Reading digits in natural images with unsupervised feature learning, *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, volume2011, p. 5.

[116] **Kalantidis, Y.**, **Pueyo, L.G.**, **Trevisiol, M.**, **van Zwol, R. and Avrithis, Y.** (2011). Scalable triangulation-based logo recognition, *Proceedings of the 1st ACM International Conference on Multimedia Retrieval*, ACM, p. 20.

[117] **Huang, G.B.**, **Ramesh, M.**, **Berg, T. and Learned-Miller, E.** (2007). Labeled faces in the wild: A database for studying face recognition in unconstrained environments, **Technical Report**, Technical Report 07-49, University of Massachusetts, Amherst.

[118] **Zhou, B.**, **Lapedriza, A.**, **Xiao, J.**, **Torralba, A. and Oliva, A.** (2014). Learning deep features for scene recognition using places database, *Advances in Neural Information Processing Systems*, pp.487–495.

[119] **Zhou, B.**, **Lapedriza, A.**, **Khosla, A.**, **Oliva, A. and Torralba, A.** (2017). Places: A 10 million image database for scene recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*.

[120] **Russell, B.C.**, **Torralba, A.**, **Murphy, K.P. and Freeman, W.T.** (2008). LabelMe: a database and web-based tool for image annotation, *International Journal of Computer Vision*, *77*(1-3), 157–173.

[121] **Turk, A.M.** (2012). Amazon mechanical turk, *Retrieved August*, *17*, 2012.

[122] **Nair, V. and Hinton, G.E.** (2010). Rectified linear units improve restricted boltzmann machines, *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pp.807–814.

[123] **Xu, B.**, **Wang, N.**, **Chen, T. and Li, M.** (2015). Empirical evaluation of rectified activations in convolutional network, *ArXiv Preprint ArXiv:1505.00853*.

[124] **Clevert, D.A.**, **Unterthiner, T. and Hochreiter, S.** (2015). Fast and accurate deep network learning by exponential linear units (elus), *ArXiv Preprint ArXiv:1511.07289*.

[125] **Klambauer, G.**, **Unterthiner, T.**, **Mayr, A. and Hochreiter, S.** (2017). Self-normalizing neural networks, *Advances in Neural Information Processing Systems*, pp.972–981.

[126] **Goodfellow, I.**, **Warde-Farley, D.**, **Mirza, M.**, **Courville, A. and Bengio, Y.** (2013). Maxout Networks, ***S. Dasgupta and D. McAllester***, *editors, Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, PMLR, Atlanta, Georgia, USA, pp.1319–1327, `http://proceedings.mlr.press/v28/goodfellow13.html`.

[127] **Ioffe, S. and Szegedy, C.** (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift, *International Conference on Machine Learning*, pp.448–456.

[128] **Boureau, Y.L.**, **Bach, F.**, **LeCun, Y. and Ponce, J.** (2010). Learning mid-level features for recognition, *Computer Vision and Pattern Recognition (CVPR), 2010 IEEE Conference on*, IEEE, pp.2559–2566.

[129] **Boureau, Y.L.**, **Le Roux, N.**, **Bach, F.**, **Ponce, J. and LeCun, Y.** (2011). Ask the locals: multi-way local pooling for image recognition, *Computer Vision (ICCV), 2011 IEEE International Conference on*, IEEE, pp.2651–2658.

[130] **Jia, Y.**, **Huang, C. and Darrell, T.** (2012). Beyond spatial pyramids: Receptive field learning for pooled image features, *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, IEEE, pp.3370–3377.

[131] **Zeiler, M.D. and Fergus, R.** (2013). Stochastic pooling for regularization of deep convolutional neural networks, *ArXiv Preprint ArXiv:1301.3557*.

[132] **Murray, N. and Perronnin, F.** (2014). Generalized max pooling, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.2473–2480.

[133] **Yu, D.**, **Wang, H.**, **Chen, P. and Wei, Z.** (2014). Mixed pooling for convolutional neural networks, *International Conference on Rough Sets and Knowledge Technology*, Springer, pp.364–375.

[134] **Gong, Y.**, **Wang, L.**, **Guo, R. and Lazebnik, S.** (2014). Multi-scale orderless pooling of deep convolutional activation features, *European Conference on Computer Vision*, Springer, pp.392–407.

[135] **Gulcehre, C.**, **Cho, K.**, **Pascanu, R. and Bengio, Y.** (2014). Learned-norm pooling for deep feedforward and recurrent neural networks, *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, Springer, pp.530–546.

[136] **Yoo, D.**, **Park, S.**, **Lee, J.Y. and Kweon, I.S.** (2015). Multi-scale pyramid pooling for deep convolutional representation, *Computer Vision and Pattern Recognition Workshops (CVPRW), 2015 IEEE Conference on*, IEEE, pp.71–80.

[137] **Graham, B.** (2014). Fractional max-pooling, *ArXiv Preprint ArXiv:1412.6071*.

[138] **Lee, C.Y.**, **Gallagher, P.W. and Tu, Z.** (2016). Generalizing pooling functions in convolutional neural networks: Mixed, gated, and tree, *Artificial Intelligence and Statistics*, pp.464–472.

[139] **Hubel, D.H. and Wiesel, T.N.** (1962). Receptive fields, binocular interaction and functional architecture in the cat's visual cortex, *The Journal of Physiology*, *160*(1), 106–154.

[140] **Sakai, K. and Tanaka, S.** (2000). Spatial pooling in the second-order spatial structure of cortical complex cells, *Vision Research*, *40*(7), 855 – 871, `http://www.sciencedirect.com/science/article/pii/S0042698999002308`.

[141] **Lin, D.**, **Lu, C.**, **Liao, R. and Jia, J.** (2014). Learning important spatial pooling regions for scene classification, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp.3726–3733.

[142] **Chollet, F.** *et al.*, (2015), Keras, `https://github.com/keras-team/keras`.

[143] **Abadi, M.**, **Barham, P.**, **Chen, J.**, **Chen, Z.**, **Davis, A.**, **Dean, J.**, **... Kudlur, M.** (2016). TensorFlow: A System for Large-Scale Machine Learning., *OSDI*, volume 16, pp.265–283.

[144] **Robbins, H. and Monro, S.** (1951). A stochastic approximation method, *The Annals of Mathematical Statistics*, 400–407.

[145] **Kingma, D.P. and Ba, J.** (2014). Adam: A method for stochastic optimization, *ArXiv Preprint ArXiv:1412.6980*.

[146] **Zeiler, M.D.** (2012). ADADELTA: an adaptive learning rate method, *ArXiv Preprint ArXiv:1212.5701*.

[147] **Wahrburg, A.**, **Zeiss, S.**, **Matthias, B. and Ding, H.** (2014). Contact force estimation for robotic assembly using motor torques, *Automation Science and Engineering (CASE), 2014 IEEE International Conference on*, IEEE, pp.1252–1257.

[148] **Ugurlu, B.**, **Nishimura, M.**, **Hyodo, K.**, **Kawanishi, M. and Narikiyo, T.** (2012). A framework for sensorless torque estimation and control in wearable exoskeletons, *Advanced Motion Control (AMC), 2012 12th IEEE International Workshop on*, IEEE, pp.1–7.

[149] **Migliore, S.A.**, **Brown, E.A. and DeWeerth, S.P.** (2005). Biologically inspired joint stiffness control, *Robotics and Automation, 2005. ICRA 2005. Proceedings of the 2005 IEEE International Conference on*, IEEE, pp.4508–4513.

[150] **Schmit, N. and Okada, M.** (2012). Design and realization of a non-circular cable spool to synthesize a nonlinear rotational spring, *Advanced Robotics*, *26*(3-4), 234–251.

[151] **Fiorio, L.**, **Parmiggiani, A.**, **Berret, B.**, **Sandini, G. and Nori, F.** (2012). pn-rVSA: human-like actuator with non-linear springs in agonist-antagonist configuration, *Humanoid Robots (Humanoids), 2012 12th IEEE-RAS International Conference on*, IEEE, pp.502–507.

[152] **Yigit, C.B.** (2018). *Novel Mechanism and Controller Design for Hybrid Force-Position Control of Humanoid Robots*, (Ph.D. thesis), Istanbul Technical University.

[153] **Narendra, K.S. and Parthasarathy, K.** (1990). Identification and control of dynamical systems using neural networks, *IEEE Transactions on Neural Networks*, *1*(1), 4–27.

[154] **Yegerlehner, J.D. and Meckl, P.H.** (1993). Experimental implementation of neural network controller for robot undergoing large payload changes, *Robotics and Automation, 1993. Proceedings., 1993 IEEE International Conference on*, IEEE, pp.744–749.

[155] **Nho, H.C. and Meckl, P.** (2003). Intelligent feedforward control and payload estimation for a two-link robotic manipulator, *IEEE/ASME Transactions on Mechatronics*, *8*(2), 277–282.

[156] **Leahy, M.**, **Johnson, M.A. and Rogers, S.K.** (1991). Neural network payload estimation for adaptive robot control, *IEEE Transactions on Neural Networks*, *2*(1), 93–100.

[157] **Byravan, A. and Fox, D.** (2017). Se3-nets: Learning rigid body motion using deep neural networks, *Robotics and Automation (ICRA), 2017 IEEE International Conference on*, IEEE, pp.173–180.

[158] **Smith, A.C.**, **Mobasser, F. and Hashtrudi-Zaad, K.** (2006). Neural-network-based contact force observers for haptic applications, *IEEE Transactions on Robotics*, *22*(6), 1163–1175.

[159] **Decety, J. and Grèzes, J.** (1999). Neural mechanisms subserving the perception of human actions, *Trends in Cognitive Sciences*, *3*(5), 172–178.

[160] **McIntyre, J.**, **Zago, M.**, **Berthoz, A. and Lacquaniti, F.** (2001). Does the brain model Newton's laws?, *Nature Neuroscience*, *4*(7), 693.

[161] **Friedman, J. and Flash, T.** (2007). Task-dependent selection of grasp kinematics and stiffness in human object manipulation, *Cortex*, *43*(3), 444–460.

[162] **Helbig, H.B.**, **Graf, M. and Kiefer, M.** (2006). The role of action representations in visual object recognition, *Experimental Brain Research*, *174*(2), 221–228.

[163] **Negri, G.A.**, **Rumiati, R.I.**, **Zadini, A.**, **Ukmar, M.**, **Mahon, B.Z. and Caramazza, A.** (2007). What is the role of motor simulation in action and object recognition? Evidence from apraxia, *Cognitive Neuropsychology*, *24*(8), 795–816.

[164] **Gupta, A.**, **Kembhavi, A. and Davis, L.S.** (2009). Observing human-object interactions: Using spatial and functional compatibility for recognition, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *31*(10), 1775–1789.

[165] **Bohg, J.**, **Welke, K.**, **León, B.**, **Do, M.**, **Song, D.**, **Wohlkinger, W.**, **Madry, M.**, **Aldóma, A.**, **Przybylski, M.**, **Asfour, T.** *et al.* (2012). Task-based grasp adaptation on a humanoid robot, *IFAC Proceedings Volumes*, *45*(22), 779–786.

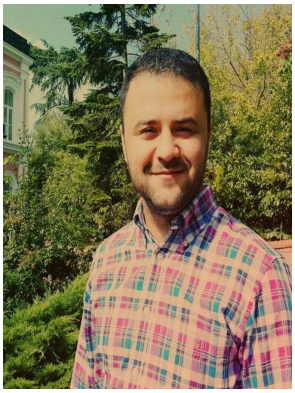
[166] **Howard, M.**, **Braun, D.J. and Vijayakumar, S.** (2013). Transferring human impedance behavior to heterogeneous variable impedance actuators, *IEEE Transactions on Robotics*, *29*(4), 847–862.

[167] **Botzer, L. and Karniel, A.** (2013). Feedback and feedforward adaptation to visuomotor delay during reaching and slicing movements, *European Journal of Neuroscience*, *38*(1), 2108–2123.

[168] **Koppula, H.S. and Saxena, A.** (2016). Anticipating human activities using object affordances for reactive robotic response, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *38*(1), 14–29.

[169] **Matsui, H.**, **Ryu, M. and Kawabata, H.** (2017). Visual Feedback of Target Position Affects Accuracy of Sequential Movements at Even Spaces, *Journal of Motor Behavior*, 1–8.

CURRICULUM VITAE



Name Surname: Ertuğrul Bayraktar

Place and Date of Birth: Adapazarı - 31/10/1985

E-Mail: ertugrulbayraktar@gmail.com

EDUCATION:

- **B.Sc.:** 2009, Yıldız Technical University, Faculty of Mechanical Engineering, Mechanical Engineering
- **M.Sc.:** 2011, Kadir Has University, Graduate School of Science and Engineering, Financial Engineering
- **M.Sc.:** 2013, Istanbul Technical University, Graduate School of Science Engineering and Technology, Mechatronics Engineering

PROFESSIONAL EXPERIENCE AND REWARDS:

- 2010-2011 Düzce University Mechatronics Engineering Department.
- 2011-2018 İstanbul Technical University Graduate School of Science Engineering and Technology Mechatronics Engineering Department

PUBLICATIONS, PRESENTATIONS AND PATENTS ON THE THESIS:

- **E. Bayraktar**, P. Boyraz, "Analysis of feature detector and descriptor combinations with a localization experiment for various performance metrics", *Turkish Journal of Electrical & Computer Sciences*, 25: 2444-2454, 5/2017. DOI: 10.3906/elk-1602-225
- **E. Bayraktar**, C. B. Yiğit, P. Boyraz, "A Hybrid Image Dataset Towards Bridging The Gap Between Real And Simulation Environments For Robotics", Under Review at *Machine Vision and Applications*.
- C. B. Yiğit, **E. Bayraktar**, P. Boyraz, "Low-Cost Variable Stiffness Joint Design Using Translational Variable Radius Pulleys", Under Review at *Mechanism and Machine Theory.*
- **E. Bayraktar**, C. B. Yiğit, P. Boyraz, “Robotic Arm Control By Fine-Tuned Convolutional Neural Network Model”, *25th Signal Processing and Communications Applications Conference (SIU*), 2017. DOI: 10.1109/SIU.2017.7960444

- C. B. Yiğit, O. Kaya, **E. Bayraktar**, P. Boyraz, “Online External Force/Torque Estimation of Antagonistic Cable-driven Flexible Joints Using Only Position Sensors”, 02/2017, International Conference on Mechatronics Systems and Control Engineering 2017, Kayseri, http://dl.acm.org/citation.cfm?id=3045714

- **E. Bayraktar**, C. B. Yiğit, P. Boyraz, “Tailoring the AI for Robotics: Fine-tuning Predefined Deep Convolutional Neural Network Model for a Narrower Class”, 02/2017, International Conference on Mechatronics Systems and Control Engineering 2017, Kayseri, http://dl.acm.org/citation.cfm?id=3045714

- H. Türkmen, N. Öksüzömer, C. B. Yigit, **E. Bayraktar**, H. Küçük, P. Boyraz “Robust and smooth colour tracking by a tendon driven humanoid neck”, 09/2016, 8th International Conference on Image Processing, Wavelet and Applications, Istanbul, Turkey.

OTHER PUBLICATIONS, PRESENTATIONS AND PATENTS:

- **E. Bayraktar**, A. H. Bilge, “Sensitivity analysis for the Markowitz model”, 05/2013, EURO Working Group for Commodities and Financial Modelling 2013, Ankara, http://ewgcfm2015.iam.metu.edu.tr/index.html

- **E. Bayraktar**, P. Boyraz, "Bir Zeplinin Otonom Hava Aracı Olarak Tasarımı, Modellenmesi ve Kontrolü", 09/2011, 560-566, TOK 2013, Malatya, http://tok2013.inonu.edu.tr/

- **E. Bayraktar**, A. H. Bilge, “Determination the parameters of Markowitz portfolio optimization model”, 07/2011, International Conference on Mathematical Finance and Economics 2011, Istanbul, http://www.mat.itu.edu.tr/icmfe2011/icmfe.html