

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

**ESKİZİ ETKİLEŞİM ŞEKLİ OLARAK KULLANAN ÖNCÜL YAZILIMLAR
ÜZERİNE BİR MODEL ÖNERİSİ**

YÜKSEK LİSANS TEZİ

Turan ALTINTAŞ

Bilişim Anabilim Dalı

Mimari Tasarımda Bilişim

Anabilim Dalı : Herhangi Mühendislik, Bilim

Programı : Herhangi Program

ŞUBAT 2015

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

**ESKİZİ ETKİLEŞİM ŞEKLİ OLARAK KULLANAN ÖNCÜL YAZILIMLAR
ÜZERİNE BİR MODEL ÖNERİSİ**

YÜKSEK LİSANS TEZİ

**Turan ALTINTAŞ
(523081016)**

Bilişim Anabilim Dalı

Mimari Tasarımda Bilişim

**Tez Danışmanı: Prof. Dr. Arzu ERDEM
Anabilim Dalı : Herhangi Mühendislik, Bilim
Programı : Herhangi Program**

ŞUBAT 2015

İTÜ, Fen Bilimleri Enstitüsü'nün 523081016 numaralı Yüksek Lisans Öğrencisi **Turan Altıntaş**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı “Eskizi Etkileşim Şekli Olarak Kullanan Öncül Yazılımlar Üzerine Bir Model Önerisi” başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

Tez Danışmanı : **Prof. Dr. Arzu ERDEM**
İstanbul Teknik Üniversitesi

Jüri Üyeleri : **Doç. Dr. Meltem AKSOY**
İstanbul Teknik Üniversitesi

Yrd. Doç. Dr. Avşar GÜRPINAR
Bilgi Üniversitesi

Teslim Tarihi : **15 Aralık 2014**
Savunma Tarihi : **21 Ocak 2015**

İÇİNDEKİLER

Sayfa

İÇİNDEKİLER	v
ŞEKİL LİSTESİ.....	vii
ÖZET.....	ix
SUMMARY	xi
1. GİRİŞ	1
1.1 Tezin Amacı ve Güdüsü	2
1.2 Yöntem	3
2. ESKİZ ve ESKİZE ÖYKÜNEN YAZILIM ve DONANIMLAR	7
2.1 Tezin Yaklaşımı İle İlişkili Olarak Eskiz	7
2.1.1 Eskiz ve ifade ettikleri hakkında	7
2.1.2 Eskiz yönteminin, bilgisayardaki çizimden farklı nitelikleri	9
2.2 Eskize Öykünen Öncül Yazılım ve Donanımlar	12
2.1.1 SketchPad (1963)	13
2.1.2 Assist (2000)	17
2.1.3 Newton platformu (1993).....	19
2.1.4 Dynamic geometry sketching (2012).....	21
2.1.5 Glassified ruler (2013)	21
3. BİR MODEL ÖNERİSİ.....	25
3.1 Eskizin Kullanım Senaryolarının Bilgisayardaki Muadillerinin Yorumu.....	26
3.1.1 Sonuca denemelerleyakınsayarak ulaşıldığı eskiz çalışmaları.....	26
3.1.1.1 Üst üste yapılan tekrarlı çizim ile tasarım senaryosu	26
3.1.1.2 Karşılıklı iki nesnenin çifter denemelerle tasarım senaryosu	28
3.1.2 İletişim veya kayıt amaçlı eskiz çalışması	30
3.1.2.1 Çizim üzerinden iletişim kurma senaryosu	30
3.1.2.2 Çizim ile kayıt tutma senaryosu	33
3.1.3 Bilgisayarda üretilen eskizin, hipermetin kavramıyla kurgulanması.....	35
3.1.3.1 Mevcut bilgi üzerine eskizle eklemeler	36
3.1.3.2 Aynı mevcut bilgi üzerine yapılan eskizlerin beraber okunması	37
3.2 Yazılım Modeli	38
3.2.1 Modelin ilkeleri	39
3.2.2 Yazılım modelinin kullanım şekli üzerinden anlatılması	42
3.2.3 Yazılım modelinin, yazılım kurgusu üzerinden anlatılması	51
3.2.3.1 Grafik nesnelerin tanımlanışı	52
3.2.3.2 İlişkili hareket kaydı	54
3.2.3.3 Zaman nesnesi	57
3.2.3.4 Arayüz nesnesi.....	59
3.3 Modelin Kullanım Örnekleri	60
3.3.1 Eş birçok parçadan oluşan cephe kompozisyonu.....	61
3.3.2 Bir kafeteryanın cephe elemanlarının plan düzleminde tasarlanması.....	62
3.3.3 Plan düzleminde üst üste denemelerle tasarım	63

3.3.4 Bir mobilya tasarımında malzeme seçimi yapılması	64
4. SONUÇ	67
4.1 Öneri Modeli Kuvvetlendirecek Teknolojiler	68
KAYNAKLAR.....	71
EKLER.....	73
ÖZGEÇMİŞ	85

ŞEKİL LİSTESİ

Sayfa

Şekil 1.1	: Hypercard grafik programlama arayüzü ve onunla yapılmış uygulama.	3
Şekil 1.1	: Tezin bağlamında eskiz ve eskizi arayüz olarak kullanan yazılımlar.....	5
Şekil 2.1	: Bazı eskiz/tasvir yöntemleri.....	8
Şekil 2.2	: Bir hayvan kafası çizimi/kullanıcının el hareketleri.....	10
Şekil 2.3	: Cad/Draft programlarımda zor yapılabilen bazı ifadeler.....	11
Şekil 2.4	: Altındaki çizimi gösterebilen saydam tablet bilgisayar.....	12
Şekil 2.5	: Sketchpad temel çizim fonksiyonları.....	14
Şekil 2.6	: Sketchpad ile birbirinin somutlanan örneği olan nesnelerin ilişkisi.....	15
Şekil 2.7	: SketchPad ile kafes giriş kuvvet analizi.	16
Şekil 2.8	: SketchPad'in kullanım alanlarından örnekler.....	17
Şekil 2.9	: Assist programında çizilmiş basit bir fiziksel deney ortamı.....	18
Şekil 2.10	: Newton işletim sisteminde, notepad yazılımına kalemle veri girişi...	20
Şekil 2.11	: Sketching Dynamic Geometry yazılımının örnek kullanımı.	22
Şekil 2.12	: Glassified Ruler çalışma şekli anlatır şema.	23
Şekil 3.1	: Vaziyet planı üzerinde konut bloğu tasarımı.	27
Şekil 3.2	: Öneri Yazılımla Vaziyet planı üzerinde konut bloğu tasarımı.	27
Şekil 3.3	: Vaziyet planı üzerinde konut yerleşimi çalışması.....	29
Şekil 3.4	: Öneri Yazılımla Vaziyet planı üzerinde konut bloğu yerleşimi.	29
Şekil 3.5	: Şantiyede haberleşme için iki farklı methodun karşılaştırılması.....	31
Şekil 3.6	: Öneri Yazılımla şantiyede haberleşme şekli.....	32
Şekil 3.7	: Bir imalat şeklinin kaydedilmesi.	33
Şekil 3.8	: Öneri yazılımla, tablet bilgisayar ile imalat kaydı.....	34
Şekil 3.9	: Öneri yazılımla, internet üzerindeki bilgiyi kullanarak kolaj-eskiz oluşturma.....	37
Şekil 3.9	: Öneri yazılımla, internet üzerindeki aynı malzemeyi kullanarak oluşturulmuş kolaj-eskizlerin beraber okunması	38
Şekil 3.10	: Baskı hassasiyetli dijital kalem ve tablet bilgisayar	39
Şekil 3.11	: Öneri modelin arayüzü ve seçim alanları hakkında sıralı ekran çıktısı örnekleri	41
Şekil 3.12	: Fare ile kullanılan masaüstü bir bilgisayar ekranıyla, parmak ile pozisyon girilen tablet bilgisayardaki yazılım arayüzlerinin kapladığı alan karşılaştırması.....	41
Şekil 3.13	: Modelin kullanıcıyla ilişki kurduğu temel öğeler	42
Şekil 3.14	: Model yazılımın kullanım şeklini anlatır tablo	43
Şekil 3.15	: Öneri yazılımın özet çalışma şekli diyagramı	52
Şekil 3.16	: Grafik nesne tanımları ve görüntüleri	54
Şekil 3.17	: Grafik nesnelerin birbirleriyle, konumlarıyla tanımlanan ilişkileri.....	56
Şekil 3.18	: Grafik nesnelerin içinde tanımlanan başka nesneler.....	57
Şekil 3.19	: Grafik nesnelerin içinde tanımlanan başka nesneler.....	58
Şekil 3.20	: Parmak ve kalem konum girdisini gruplayan / yorumlayan nesneler...	59
Şekil 3.21	: Ekranda değilen noktada çıkan menülerin açıldığı alanlar.....	60
Şekil 3.22	: Üçgen cephe kompozisyonu.	61
Şekil 3.23	: Kafeterya cephesi tasarımı.....	62
Şekil 3.24	: Üst üste denemelerle tasarım.	63
Şekil 3.25	: Mobilya için malzeme seçimi.	65
Şekil 4.1	: Tablet üzerinde cetvel ve elektroforik ekran önerileri.....	69

Şekil 4.2 : Kolaj, İnternet üzerinden bir kelime ile aratıldığında çıkan resimli sonuçların, bu resimlerin internetten alıp kullanan elektronik eskizlerle beraber arama motorunda gözükmesi.	70
---	----

BİLGİSAYARDA, ESKİZİ ETKİLEŞİM ŞEKLİ OLARAK KULLANAN ÖNCÜL YAZILIMLAR ÜZERİNE BİR MODEL ÖNERİSİ

ÖZET

Bugün, yazılım ve bilgisayarlar, günlük hayatta yapılan haberleşme, çizim yapma, bir hiyerarşideki bilgiye ulaşma, hesaplama yaptırma gibi belirli bir yolu olan tekrarlı işler yanında, arkadaşlarla buluşma ayarlama, fotoğraf çekme, internet üzerinde arama yapma gibi, katagorilendirmenin veya isimlendirmenin anlamlı olmayacağı, hassasiyet kaygısı olmadan ve az aşamalı işlerde de kullanılmaya başlamıştır. İnternete her yerden ulaşılabilmesini sağlayan teknolojik altyapı ve akıllı telefon gibi donanımlar, bilgisayarın her yere taşınabilen ve üzerinde karmaşık olmayan rutinlere sahip işler yaptırılabilir bir araç kutusu haline getirmektedir. Bu tezin amacı bu basit araçlar kutusuna, yine günlük hayatta çokca kullanılan kalem ve kağıdı da, “ilişkili bilgi kaydetme” başlığı altında yorumlayıp bir yazılım biçiminde eklemektir.

Eskizin birçok kullanım alanından birisi, düz yazı yazma veya bir şeyin resmini çizme gibi sonlu eylemlerin yanında, bir işe başlarkenki ilk planı yapmak, bir ortamda çalışırken o ortamın sunamadığı basit bir soyutlamayı çabuk şekilde kayıt altına almak gibi başka işlerle alakalı yardımcı bir araç olarak kullanılmalıdır. Örnek vermek gerekirse, bir müzisyen bir kayıt yaparken hoşuna giden bir tekrarın kaç tane olunca güzel olduğunu, o güzel olduğu şekilde kaydetmeden önce bir yere not alır, veya bir tasarımcı bir bardak tasarlarken aklına gelen ilk şekilleri hızlı biçimde çizer, daha sonra onları başka bir ortamda uzun uzun detaylandırır.

Bu çalışma bu açıdan eskize bakıp, bu kabiliyetinin hangi bilgisayar fonksiyonlarıyla nasıl daha kuvvetlenebileceği hakkında öneriler yapmaktadır. Bunu yaparkenki kaygısı bu fonksiyonlara ulaşım şekillerinin, insanla eskiz arasındaki kesintisiz ve aşamasız direk ilişkisinin sekteye uğratılmamasıdır.

Tez metni boyunca eskizin yukarıda bahsi geçen çerçevedeki nitelikleri anlatılacak, kağıt kaleme öykünen daha önceden yapılmış yazılımlar incelenecek, sonrasında çeşitli eskiz kullanım senaryolarında, nasıl yazılım fonksiyonlarının bu eylemleri nasıl kolaylaştırabileceği hakkında fikirler aktarılacaktır. Daha sonrasında bir yazılım modeli, kullanım şekli ve yazılım kurgusu anlatılarak önerilecek, bununla yapılabilecek çeşitli eylemler anlatılacaktır. Bu anlatım sıralı ekran görüntüleriyle, bazı fonksiyonların algoritmalarının anlatımıyla ve olası kullanım senaryolarının anlatılması yoluyla yapılacak, sonuç bölümünde de bu önerinin ne kadar amaçladığı noktaya ulaştığı sorgulanacaktır.

A NEW MODEL BUILT ON PREDESSOR SOFTWARES THAT USES SKETCH METAPHOR AS A USER-COMPUTER INTERACTION

SUMMARY

Recent form of computers as tablets and smart phones lets users to use them as a tool for simple tasks that would not require precision and results in a strict medium, such as as take a photo of interesting stuff and ask a friend on social network if he/she knows what it is. These are not tasks such as drafting buildings technical drawing or engineering calculations, or stock control of a warehouse. And these tasks come up with wide usage of internet almost in everyplace and technology that make computers so small and energy-efficient, in a simpler phrase 'mobile'. Today we have a mobile toolset, and this should be useful for practical kind of uses, and there are lots of that tools in smart phones and tablets.

Sketch is also something mobile. In this thesis, sketch is considered as a medium to help what we are doing or thinking, with allowing us to build visual structures, rather than writing down something or making a drawing of a scene. This kind of usage of sketch is very common, and most of the time when something comes up to our minds, or if we want to tell something too complicated to express with words, we look for paper. The aim of this study is to look for new ways to strengthen this use of sketch, by proposing new functions in software, but without interpreting the natural and direct way of sketching. Scenarios of this(mentioned above) use of sketch are analysed and new tools are proposed to root for a new way of expressing or exploring what is in our minds in these scenarios, to the screen of tablet computer with a electronic pen and fingers.

In first chapter, what is the description of sketch and software with sketching metaphor are mentioned, from thesis' perspective. Sketching is supposed to be a tool for content creation with symbols. And software with this metaphor supposed to be a content creation with those symbols on a tablet computer, which have some computer-wise attributes, different than symbols drawn on to physical paper. Also survey method of some usage scenarios of paper in content creation which are examples of subjects of this thesis.

The second chapter of this study is a review some past software and hardware proposals which have a unique reassembling of user-computer interaction with a metaphore of sketching. None of those software (like Sutherland's SketchPad) are designed to mimic well known pen-paper sketching, rather they propose new way of interaction using the metaphore of skething . So this thesis study is not looking for a better way of sketching in computer, it looks for using sketch as a medium of software control. Review which is mentioned above involves user-computer interaction examples which are explanied with usage steps of this software in different scenario, and also short explanations of aim of those studies. Usage steps of those softwares are explained with step by step illustrations of computer screen and a description below,

which will be also common way of expression within all chapters of thesis. Thesis also mentions about properties of pen-paper sketching which has a potential to transform into a human-computer interaction and properties that makes sketching still a indispensable tool for us and should not be abandoned in favor of more software functions. By the way it should be mentioned that in this thesis, the content of sketching always thoughted a be a medium for analog data structures such as a shopping list with prices, places to buy and personal notes, rather than a beautiful and realistic freehand sketch of a tree. It should be highlighted that those software which are reviewed in second chapter have this kind of perspective onto sketching metaphor.

In third chapter some work scenarios about content creation with physical sketch and widely used computer softwares will be compared. Those softwares are not softwares mentioned in second chapter which use sketch as a metaphor .They are in this comparison because they are softwares which comes up to mind first to accomplish related scenario. This comparison will be about steps of progress and elapsed time to finish the job of related scenario. Than this two output will be illustrated and some distinct points of differences will be listed. After this comparision, a new software routine which benefits both from sketch and software in comparison, will be proposed. Those proposals will be fictional software routines which work on a tablet computer with pen and touch input. And aim of those proposals are to create a fusion of good sides of both physical sketching and computer which are revealed in comparison. After this comparisons and proposals, a fictional software proposal which consists all of those past-proposed software routines and more will be presented. Some fragments of this fictional software is also coded (code is presented in the end of thesis) with Java to demonstrate some fragments with real interaction. This fictional model software will be explained with step by step illustrations of usage, explanations of algorithms/routines of software and possible working scenarios with it.

In last chapter software proposal will be evaluated of how much it succeed its motive, and downsides/potentials will be explained. This evaluation will be about how much it succeed its aim, proposing a collection of computer functions which will useful for content creation, in a way that won't restrain the practicality and freedom of pen-paper sketching. Also new possible technologies and usage scenarios will be mentioned, which would be helpful for proposal, to achieve its motive.

After mention about structure of thesis and its goal, it is also worthwhile to explain how does proposed software works, thus illustrations of fictional screenshots and diagrams in thesis which are not shown in this abstract, are very important elements to describe it. The software have no selection menu or array of buttons, so when this sketch software started, there is only white blank screen, just like piece of paper. User have two tools for input, first one is a digital pen that lets user draw shapes on screen. Digital pen always draw shape on screen in every phase and moment in this content creation. Users can draw anyplace in screen just like sketching with pen and paper because there is no reserved area for interface or something else. Second way of input is touching screen with fingers. When screen is touched, a floating and transparent menu appear on the touched area of screen. This menu is responsive to the shapes and graphic object under touched area. If user touches a blank area, a creation menu will appear, if user touches on a graphic object or a shape, a menu to edit this object pops up. User make selection with sliding his/her finger on those menus. Some of functions listed on those menus may require a second finger input, or require pencil input. With this interface user can add graphic objects which can effect what he/she draws, or construct something with those objects. To clarify difference between freehand shapes that

purely created with digital pen and graphic objects made with help of finger touch functions; freehand shapes can not interact and they are as easy as physical sketching but graphic objects can interact within themselves and freehand shapes. But this interaction requires a definition which can be constructed with pop up menus, finger movements and selections. There are five kinds of graphic objects; point, connected points, freehand with reference point, aperture with reference point and time objects. All of those objects have position attribute. Users can define a connection within position attributes of different objects, so when a point of those objects moves, other one can move within this definition. Definition can be created by moving those objects at the same time, and software records those movement changes and create a relation between them. Point object is a bare point which has just position attribute. Connected points are polylines with position attribute and color attribute. Freehand sketch with reference point has color and reference point position attribute. Color attributes are also presented as points attached to reference points of those objects and they can be connected to another objects position attribute. Different than those three graphic objects there is a time object which records all history of sketch and let user to see different moments and fragments of their work. Time object is a line with points which is drawn by combination of digital pen and finger input on screen. User controls which fragment of sketch appears in screen by moving control points with finger touch. Another object is aperture with a reference point, which is actually a user drawn hole in sketching canvas in order to see another software (just like an internet browser, a video player etc.) through it. User can draw something on it with electronic pen or, use the software through this hole with finger touch. All those five objects can be copied as instances, can be deleted, can be connected or can be redefined.

To sum up the goal of the proposed software, it intends to create a digital tool kit for content creation, which is similar to content creation with physical sketch. Without any intervention to the behavior of physical sketching as possible, proposal tries to mimic sketching in a tablet computer and add some software functions which also can be created and controlled with sketching metaphor. To achieve this, firstly thesis examined some picked out software that use sketching as a metaphor for user interaction, then used some methods of doing things from those software. Then some working scenarios which are typical examples of content creation with physical sketching are examined, then new scenario specific softwares, which try to achieve an efficient hybrid between sketch and potentials of computer, are proposed. All those past examples and new proposals combined in a fictional model software. So this software in a tablet computer can be used exactly same way as a pen and a piece of paper. In addition to this sketch simulation, users can add special functions which help them for content creation if they want, in a transparent and floating user interface which has a sketchy way of doing things.

1. GİRİŞ

Bugün çoğu işin, araştırmanın ve öğrenmenin bilgisayarlar üzerinde yapılması, bilgiyle iletişim kurma şeklimizi değiştirmiş, fakat bize gelen bu bilgiyi kurgulamanın erken aşamalarında, kağıt-kalem kullanımının yerini alamamıştır. Günümüzde de bilgisayardan birşey öğrenildiğinde, öğrenilenlerden yeni birşey oluştururken çoğu kişi ilk önce kağıt ve kalem kullanmakta, yapacağı şeyi belirledikten sonra bilgisayarda yapacağı işe uygun bir yazılımla işine devam etmektedir.

Günümüzde yazılımlar genel olarak, çeşitli iş dallarındaki belli görevlerin, bir kullanma senaryosu üzerinden kurgulanarak, daha hızlı, düzgün ve detaylı şekilde yapılmasına yarayan araçlardır. Bir metin yazılacaksa bunu en hızlı ve düzgün yazılmasını, bir ses kaydedilecekse bunu kendi senaryosu içinde kullanıldığında en hızlı ve düzgün kaydedip manipölasyon yapılmasını hedefleyen, birçok yazılımcının çok ciddi vakit vererek hazırladığı, evrilen ve giderek mükemmellik noktasına yaklaşan, çeşitli iş senaryoları için birçok yazılım mevcuttur.

Bahsi geçen bilgiyi kağıt üzerinde yazı ve şekillerle ifade bir kurgu-ürünün ve bunun daha sonrasında bilgisayar yazılımlarıyla detaylandırılması tespitine tasarım alanından örnek verilebilir. Tasarım alanında kullanılan programlar da daha önceki tasarım aşamalarının senaryoları üzerinden alınan metaforlarla tasarlanmış yazılımlardır (Oxman, 2008). Bugün mimarlık/tasarım pratiğinde tasarımın ilk aşamalarında, büyük bir çoğunluğun tasarım problemlerinin çözüm yollarını kararlaştırıldığı ortam, maket ve el çizimi gibi tasarımcıların kolay kullanabildiği ve aşına olduğu yollar olmaktadır. Bu büyük çoğunluk, tasarımın ileri aşamalarında ise, hassas ve karmaşık çizimleri yapabilecek kabiliyetteki teknik çizim¹ yazılımları kullanmaktadır. Teknik çizim yazılımları genelde sonuca ulaşmak için, tasarımın erken aşamasındaki tasarımcının belirlemek istemediği hassasiyette bilgi istemektedir, ve zaten hassasiyete ihtiyaç duymayan bir evrede tasarım yapmakta olan tasarımcı bu yazılımları bu yüzden tercih etmemektedir (Davis, 2007).

1 Teknik Çizim Yazılımları ifadesiyle belirtilmek istenen AutoCad, Revit,ArchiCad vb. kendi konusunda hassas ve kesin ürünler ortaya koyabilen drafting yazılımlarıdır.

Tasarım için teknik çizim yazılımlarının dışında , çeşitli formüller/parametrelerle ile karmaşık/kontrollü biçimlerin ve ilişkilerin üretilebildiği yazılımlar, veya bir yüzey üzerinde basınç hassasiyeti olan kalem kullanılmasıyla bilgisayar ekranında aynı gerçekteki gibi boya-kağıt ile resim çizmeye (*Wacom* tablet ile *Photoshop* yazılımında resim çizmek gibi) yarayan yazılım ve donanımlar gibi birçok farklı ve çeşitli sistemler de mevcuttur. Çeşitli formül ve parametrelerle biçimin kurulduğu tasarım araçlarını kullanmak için ciddi yazılım bilgisi gerekir, tablet sistemler ise sadece çizme eylemini taklit edip, bilgisayarın zekasını¹ kullanmaz. Tez çalışmasında, yazılımların insan-bilgisayar etkileşim prensipleri, arayüz kurgusu gibi konulara değinilip, kullanıcı tarafından az bir bilgiyle kolay kullanılabilir (eskize benzemesi üzerinden tanımlanacak bir kolaylık) ve bilgisayarın imkanlarını bu basitliği engellemeyecek şekilde işin içinde olabileceği bir sistem oluşturulmaya çalışılmıştır.

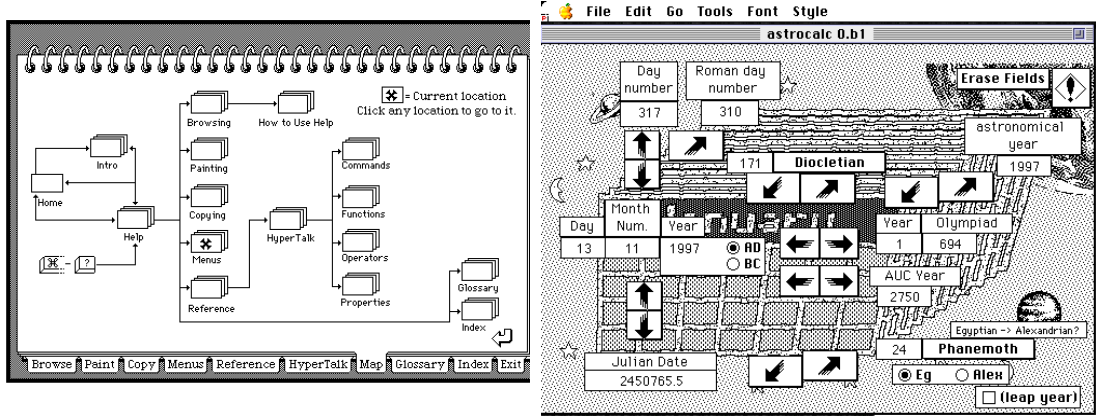
1.1 Tezin Amacı ve Güdüsü

Tezin amacı tasarımın fiziksel ortamda ilişkisel-diyagramatik düşünmemizi kolaylaştıran eskiz tekniğinin olabildiğince niteliklerini kaybetmeden tutulup, bilgisayar ortamında insan-eskiz etkileşimini en az perdeleyecek bir arayüz kurgusuyla, bilgisayarın yeteneklerini sürecin içine nasıl katılabileceğini, bilgisayar-insan etkileşiminin gerçek potansiyellerini kullanma heyecanı içinde cevaplamak isteyen bir yazılım modeli ortaya koymaktır .

Bilgisayar bilimleri tarihine bakıldığında, yazılımla uğraşan birçok mühendisin ve akademisyenin, bilgisayar kullanıcısının bir iş için özelleşmiş yazılımlar yanında çeşitli yazılım fonksiyonlarının ilişkilendirilerek kurulduğu, basit programlama dilleri/ortamları denebilecek yazılımlar ile de uğraştığı görülebilir. Böyle bir uğraşının güdüsü bilgisayarın sadece önceden tarifli işleri değil, kullanıcının kendi uydurduğu bir işi de kendi istediği şekilde bilgisayara anlatıp yaptırabilmesinin, bilgisayar kullanımında açacağı yeni potansiyellerdir (Kay, 2006). Bilgisayar bu bakış açısıyla çeşitli işleri yapan küçük modüller barındıran, kullanıcının da bu küçük modüllerle bir kompozisyon kurarak daha büyük ve farklı bir araçlar yapabilmesine olanak veren bir ortam olarak tarif edilebilecektir.

1 Zeka ile anlatılmak istenen bilgisayarın daha önce tanımlanmış işleri, kendiliğinden/kontrollü şekilde yapmasıdır

Yazılımcılar bu potansiyeli görmüş ve bunu gerçekleştirmek için SmallTalk, Hypercard, Max/Msp gibi basit nesnel programlama dilleri, grafik arayüzlere sahip programlama ortamları yapmışlardır (**Şekil 1.1**). Bu tezin güdüsü bir iş senaryosu üzerinden bir yazılım modeli ortaya koymak değil, bir bilgisayar-insan etkileşim şekli üzerinden tarif edilen bir araç seti önerisi ortaya koymaktır. Daha farklı tarif etmek gerekirse bu bakış açısına göre yazılım, bir “x” işini yapmak için değil, bir “x” iş yapma şeklini kullanıcısına sağlayan bir yazılım modeli, bir araç ortaya koymaktır.



Şekil 1.1 : Hypercard’ın grafik programlama arayüzü ve bununla yapılmış program

1.2 Yöntem

Bu tez boyunca, eskiz metaforunu insan-bilgisayar etkileşim şekli olarak kullanan bir görsel içerik oluşturma yazılımı modeli sunulacak, bu kurgu yazılım ortaya konmadan önce de fiziksel eskizin nitelikleri ve eskize benzer etkileşim şekli olan yazılımlar incelenecektir. Yine öneri model sunulmadan önce, yazarın seçtiği bazı iş senaryolarında eskizin ve drafting yazılımlarının içerik oluşturulurken kullanımının gözlemlenmesine dayanan sonuç ürün ve süreç karşılaştırılması yapılacak, bunlar hakkındaki değerlendirmeler üzerine yazarın bu iş senaryoları için kurduğu yeni hibrit yazılım önerilerinden bahsedilecektir. Daha sonrasında öneri yazılım modeli sunulacak, bu yazılım modeli ile yapılabilecek örnek iş senaryolarından bahsedilecektir. Sonuç bölümünde ise bu yazılımın değerlendirilmesi yapılacak ve potansiyellerinden bahsedilecektir. Tezin iskeleti hakkındaki bu kısa açıklamadan sonra, aşağıdaki paragraflarda tezdeki bölümlerde anlatılan konuların ele alış ve anlatım şekilleri, kavramların bu tez kapsamındaki sınırları ve yapılan gözleme dayalı değerlendirmelerin yöntemleri anlatılacaktır.

İkinci bölümde model yazılım önerisinin ortaya çıkmasında yol gösterici olan fiziksel eskizin nitelikleri ve eskiz yapma eylemini kendini kullandırtma yöntemi olarak kullanan yazılımlar incelenmektedir. Tezde bahsi geçecek fiziksel eskiz kavramının çerçevesi, eskiz ile bilginin, iki boyutlu düzlem üzerindeki biçimlerle soyutlanıp, bu ortamda yaratımı ve kaydedilmesi ile sınırlıdır. Bunların dışında eskiz, kullanıcının kafasında çok net olan veya gözüyle gördüğü bir görseli çizmesi için kullanabileceği bir araçtır, fakat bu nitelikleri ele alınmamıştır (**Şekil 1.2**). İncelenen öncül yazılımların çerçevesinde de bir sınırlamaya gidilmiştir. Bu yazılımların seçilme nedeni, herbirinin eskiz metaforunu çeşitli şekillerde insan-bilgisayar etkileşiminde kullanmalarıdır. İncelenen yazılımlar, bugün tablet bilgisayarlarda çok kullanılan ve eskizi bilgisayar üzerinde gerçekleyen yazılımlardan değil, eskiz metaforunu farklı fonksiyonlarının kullanımı için kullanan, sonuç ürünü fiziksel eskizdeki gibi bir imaj değil, programdan programa farklılık gösteren çıktılar olan yazılımlardan seçilmiştir (**Şekil 1.2**). Bu yazılımların çıktısı bir giriş üzerindeki yapısal yük diyagramı da olabilir, bir fizik deneyi simülasyonu da olabilir.

Üçüncü bölümde yazarın kendi günlük bilgisayar kullanım tecrübeleri üzerine belirlediği, bilgisayarın içinde olduğu bir iş akışının ortasında fiziksel eskizin çok daha hızlı ve etkili sonuçlar verdiği iş senaryoları sunulacaktır. Bu senaryoların, iş akışının bilgisayar ile kesintisiz devam ettirildiği haliyle ve fiziksel eskizin kullanıldığı haliyle sonuçlanan ürünleri sunulacak, süreç ve sonuçlar gözlemlenecektir. Burada yöntem olarak metin yazarı kendi yaptığı senaryoya bağlı uygulamayı, kendisi ölçmekte ve değerlendirmektedir (Pedgley,2007). Burada belirtmekte fayda var ki bu karşılaştırmada kullanılan bilgisayar yazılımları, ikinci bölümde anlatılan eskiz metaforunu kullanan yazılımlarla aynı çerçevede değildir (**Şekil 1.2**). Bu yazılımlar, ilgili senaryolarda yazarın kendisinin geçmişte kullandığı ve çok kullanıldığını gördüğü yazılımlar olduğu için, seçilmiştir. Araştırma örnekleme olarak böyle bir konu seçiminin (bilgisayarın içinde olduğu bir iş akışında fiziksel eskizin tercih edildiği iş parçaları senaryoları) nedeni, görsel içerik üretimi yapan yazılımlarla çalışılırken sadece arayüzleri yüzünden arada eskiz ortamına atlanmasının, eskizi arayüz olarak kullanacak bir yazılımların ne işe yarayabileceği hakkında çarpıcı bir başlangıç noktası verebilmesidir.

Üçüncü bölümün ikinci kısmında, tez boyunca bahsi geçen amaç ve kavramlar ışığında kurgulanmış bir yazılım önerisi anlatılmaktadır. Bu anlatım yazılımdaki fonksiyonların kullanımının anlatıldığı sıralı ekran ekran görüntüleri ile yapılmaktadır. Bu sıralı ekran görüntüleri, kullanıcı girdisi sonucu yazılım arayüzünde neler değiştiği ve girdinin ne sonuç verdiğini anlatan, altında yazılı açıklamalara sahip kurgu resimlerdir. Bu görüntüler yazılımın gerçek ekran görüntüleri değildir, zira yazılımın gerçek kodu kısmen yazılmıştır ve tüm fonksiyonları istenildiği gibi çalışmadığından bu ekran görüntüleri konmamıştır. Daha sonraki kısımda yazılımın bazı fonksiyonlarının algoritması metin, şema ve kısmi bilgisayar koduyla açıklanmıştır. Bölümün en sonunda ise sıralı ekran görüntüleriyle yazılımın çeşitli iş senaryolarında nasıl kullanılabileceği anlatılmıştır.

Sonuç kısmında üçüncü bölümün ikinci kısmında önerilen yazılımın, yine bu kısımda bahsi geçen ilke ve amaçlarını ne kadar gerçekleştirebildiği yazar tarafından değerlendirilmiş, gerçekleştirdiği ve gerçekleştiremediği noktalara değinilmiştir. Daha sonrasında yazılımı, amaçlarını gerçekleştirmesine yaklaştıracak olası teknolojiler ve yeni kullanım şekillerinden bahsedilmiştir.



Şekil 2.1 : Tezin bağlamında eskiz ve eskizi arayüz olarak kullanan yazılımlar

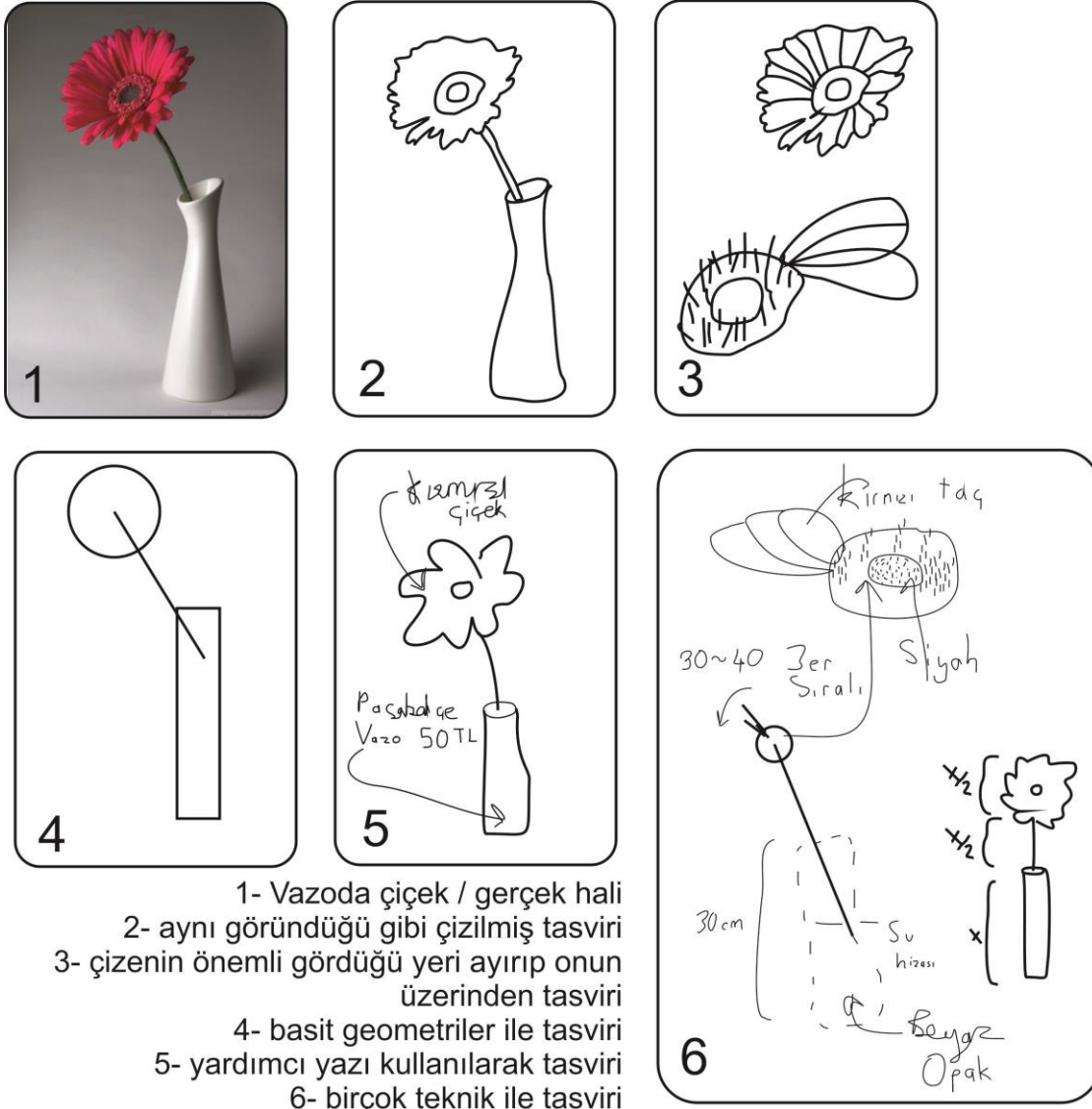
2. ESKİZ VE ESKİZE ÖYKÜNEN YAZILIM VE DONANIMLAR HAKKINDA

2.1 Tezin Yaklaşımı İle İlişkili Olarak Eskiz

Kağıt üzerine kalemle eskiz çizmek, tasarımın ilk aşamalarında, bilgisayardaki benzer işi yapabilecek araçlara göre, tasarımcılar tarafından daha çok tercih edilen bir yöntemdir. Tasarımcıların bahsedilen bu iki yöntemi kullanarak yaptıkları işlerde, eskiz yöntemi kullanan tasarımcıların, bilgisayardaki *teknik çizim* araçları kullanan tasarımcıya göre çoğunlukla daha fazla tasarım alternatifi ortaya çıkardığını gösteren araştırmalar mevcuttur (Goel, 1995). Bilgisayar ile birçok farklı tasarım alternatifi üretebilen, biçim ve aralarındaki ilişkileri değişkenler ve formüller ile bağlayan tasarım araçları (Grasshopper, Generative Components vb.) bu kıyaslamanın dışında tutulmuştur zira bu araçları kullanmak ciddi bir bilgi birikimi gerektirir ve tasarım pratiğindeki büyük çoğunluk tarafından kullanılmaz. Eskiz ile *teknik çizim* araçlarına göre erken tasarım aşamalarında daha verimli çalışılmasına sebep olarak tasarımcının, eğitimi sırasında eskiz tekniğini çokca pratik etmesi, bu iş üzerinde bilgisayar tecrübesinin daha az olması gösterilebilir. Fakat bu aradaki farkın sebebinin ne olduğunu anlamak için eskiz tekniğinin, şu anki mimarlık pratiğinde kullanılan teknik çizim araçlardan farklı kendine has özelliklerinin irdelenmesi gerekir. Tez metninin bu kısmında eskiz tekniğinin teknik çizim araçlarıyla arasındaki bu farklılıklar anlatılacaktır.

2.1.1 Eskiz Tekniği ve İfade Ettikleri Hakkında

Tez metni boyunca “eskiz” ile tarif edilecek olan eylem, bir yüzey üzerine iz bırakarak (kalem ile kağıt üzerinde bir şeyler çizmek gibi), kısa süre içerisinde (dakikalar ile tarif edilebilecek) yapılan, çeşitli kurgular altında bilginin kayıt altına alınması anlamında kullanılacaktır. Bu kurgular birçok şekilde olabilir, aşağıda vazo içerisindeki bir çiçeğin farklı eskiz kurgularıyla ifadeleri anlatılmaktadır (**Şekil 2.1**).



- 1- Vazoda çiçek / gerçek hali
 2- aynı görüldüğü gibi çizilmiş tasviri
 3- çizenin önemli gördüğü yeri ayırıp onun üzerinden tasviri
 4- basit geometriler ile tasviri
 5- yardımcı yazı kullanılarak tasviri
 6- birçok teknik ile tasviri

Şekil 2.1 Bazı eskiz/tasvir yöntemleri

- 2 numaralı eskizde, somut bir nesne gözün gördüğü gibi kaydedilmiştir (vazodaki bir çiçeğin resmini çizmek gibi)
- 3 numaralı eskizde, somut bir nesnenin aklımızda kalan önemli tarafları kaydedilmiştir (vazodaki çiçeğin güzel taç yaprakları gibi)
- 4 numaralı çizimde somut nesnelerin birbirleri arasındaki ilişkiler basit biçimler kullanılarak kaydedilmiştir (çiçeğin sapının vazo içerisindeki duruşunun bir çizgi ve dikdörtgen ile ifade edilmesi)

- 5 numaralı çizimde, somut nesnenin bazı öğelerinin yazı ile ifade edilip, kaydedilmiştir (çiçeğin taç yapraklarının bir tanesini çizip yanına rengini ve sayısını yazmak gibi)
- 6 numaralı çizimde vazo ve çiçek, tüm bu yukarıdaki kurguların karışık olarak kullanıldığı hibrit bir kurgu ile kaydedilmiştir.

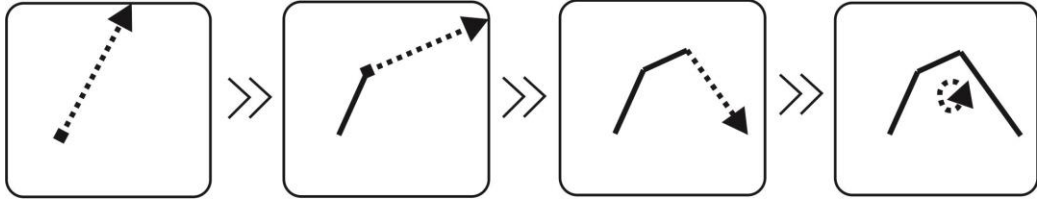
Yukarıdaki örneklerde bahsedildiği gibi eskiz, çok somut şeyler için kullanılabileceği gibi çok soyut ilişkileri de üzerine işleyebildiğimiz, bunları karma olarak kullanabildiğimiz bir tekniktir. Herkesin kendine göre zihnindekileri eskize dökme şekli vardır ve eskizde çizilen şey herkes tarafından anlaşılabilir (bir disiplin içeren) bir şey olabileceği gibi, sadece onu yapan tarafından anlam ifade edecek bir şey de olabilir. Eskizi portre yapan bir ressam da , bir doğa olayını ve alt ilişkilerini gösteren bir fizikçi de kullanabilir.

2.1.2 Eskiz Yönteminin Bilgisayardaki Çizim Yöntemlerinden Farklı Nitelikleri

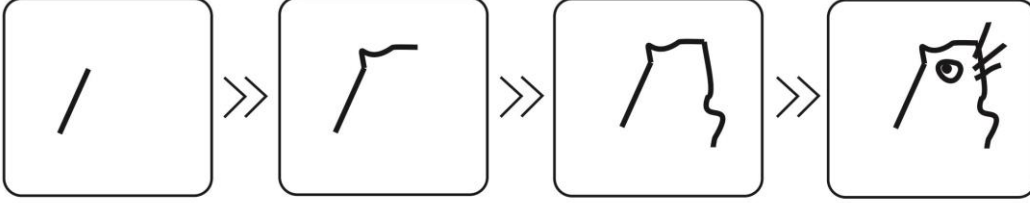
Eskiz tekniğinin bugünkü mimarlık disiplininde kullanılan teknik çizim programlarından, kullanıcının araç ile iletişim şekli bakımından, farklılıkları bulunur. Bu farklılıklar sonucu, eskiz yöntemi ile tasarım alternatifleri, bilgisayardaki teknik çizim araçlarından daha hızlı ve çeşitli biçimlerde üretilebilir (bu genelleme, tasarım dünyasının, ciddi yazılım bilgisi sahip, yazılımlardaki *script* lere, biçim ilişkilerine hakim olabilen küçük kısmı için geçerli değildir).

Bu farklılıklardan birisi eskiz üzerinde çizdiğimiz şeylerin her anını görüp, o anda müdahale yapabilmemizdir (zihinde bütün detaylarıyla beliremeyen bir biçimin, çizildikçe detaylarının ne olacağının karar verilmesi gibi- **Şekil 2.2**). Bu nitelik varılacak sonucun başlangıçta muğlak olduğu tasarım pratiğinde önemlidir. Zira tasarım hakkında yapılan araştırmalar tasarımcıların, devamlı tekrar eder bir şekilde kağıt üzerindeki bakıp ilerlemesi biçiminde tasarım yaptıklarını önerir.(Schön and Wiggins, 1992). Tasarımcı ile tasarımın yapıldığı ortam arasındaki çift taraflı iletişim, eskiz üzerinde çizme, değişiklik yapma ve yeniden algılama, devamlı olarak tekrarlanan bir döngüdür (Prats and Garner, 2006).

Teknik çizim tasarım yazılımlarında ise çizilecek biçimi kesin tarif ederiz ve tarif sonucunda kesin sonucu görürüz. Çok doğru ve hassas çizim yapabilmeyi hedefleyen programların bu nitelikleri, hassaslık gerektirmeyen ön tasarım aşamasında bir dezavantaja dönüşür.



Kullanıcının harekete başlamadan önce kafasındaki doğrultular/el hareketleri



Kullanıcının kağıt üzerinde gördüğü imajdan gelen geri bildirimle elini titretmesi, çizgiyi sonlandırması sonucu ortaya çıkan eskizler

Şekil 2.2 Bir hayvan kafası çizimi/kullanıcının el hareketleri

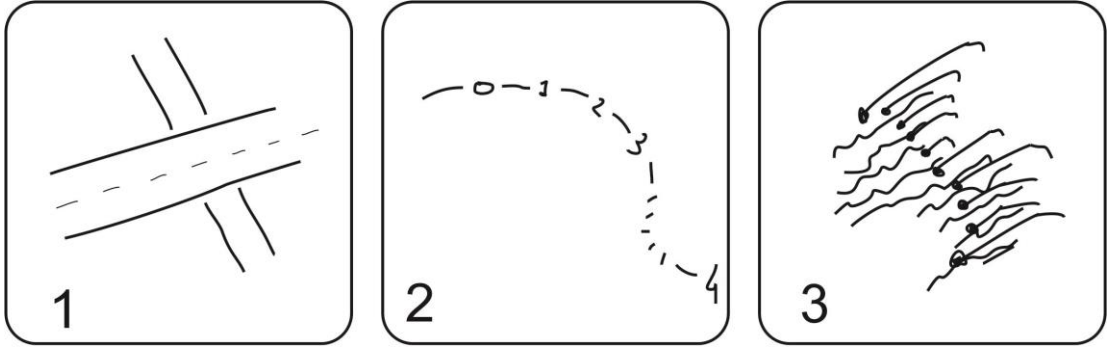
Bir diğer farklılık ise eskiz üzerindeki biçimlerin gerçekte ne tariflediklerinin muğlak olmasıdır. Örnek vermek gerekirse, bir mimar kağıt üzerinde bir ev planı çalışırken çizdiği çizgi cam da olabilir duvar veya döşemedeki bir kot farklılığı da olabilir. Çizim ilerledikçe bu çizgilerin kazandığı anlam değişebilir. Eskizde biçimlerin soyutlanmış fiziksel karşılıkları , tasarımcı tarafından anlamı değiştirilebilir ve muğlak olması, tasarım probleminin çözüm kümesini de dinamik olmasını sağlar (Purcell and Gero, 1998). Kağıt ve kalem kullanarak tasarımcı çok hızlı bir şekilde denemeler oluşturur ve bunlara bakarak ilişkiler ve olasılıklar görür, ve bu gördükleri eskizin ileri aşamalarında fikrin şekillenmesine ve yeni biçimlerin ortaya çıkmasına sebep olur.

Eskiz ortamındaki girdi *modeless*² ve tek tip bir bilgidir. Teknik çizim yazılımlarda ise çeşitli *mode*¹ lar üzerinden girdiler ile çizim oluşur. Mesela bir duvar saati çizerken önce daire aracı seçilip daire çizilir sonrasında çizgi aracı seçilerek akrep yelkovan çizilir, daha sonrasında ise yazı aracı seçilip rakamlar yazılır (bu örnekte 3 *mod* ile çizim yapılmıştır). Serbest el eskizde ise böyle bir girdi şekli çeşitlemesi yoktur

1 *mode* yazılım arayüz tasarımında, kullanıcının çeşitli araçlarla girdiği girdilerin, yazılımda nasıl yorumlanacağıdır. Örnek vermek gerekirse autocad yazılımında, daire çizmek için, daire aracının seçilmesi gerekir, daire aracı seçilirken daire dışında birşey çizilemez. Bu aracı seçerek bir *mode* 'u seçmiş oluruz.

2 *modeless* yazılım arayüz tasarımında, kullanıcı girdilerinin, herhangi bir *mode* değişikliğine ihtiyaç duymadan yorumlanmasıdır. Örnek vermek gerekirse, İpad kullanırken ekrana tek sefer değmemiz, art ardına iki kere değmemiz, birkaç parmakla değmemiz, hareket ettirmemiz, yazılımca farklı komutlar olarak yorumlanır. Bunları yapabilmek için özellikle bir *mode*'u seçmemize gerek yoktur.

(buradaki eskiz eylemiyle pergel ve yazı şablonu ile çizilen düzgün teknik el çizimi kast edilmemiştir, zira bilgisayar kullanımından önce kalem ve kağıt ile yapılan teknik el çiziminde de buna benzer *mod* lar bulunur). Eskizin *modeless* olmayan yapısı nedeniyle karışık çeşit ifadelerle sahip çizimler kolay ve hızlı şekilde yapılabilir. Birbirine değmeyen çizgiler, yazı-çizgi karışık veya nokta-eğri-çizgi karışık ifadeler (**Şekil 2.3**) eskizde zahmetsizce çizilebilir.



Şekil 2.3 Cad/Draft programlarında zor yapılabilen bazı ifadeler. 1; planda altta üstte kalan elemanlar . 2; metin ve çizginin beraber kullanıldığı hibrit ifadeler . 3; belirsiz başlayıp çizdikçe belli olan ifadeler

Bir diğer önemli nitelik ise eskiz ile yapılan işlerin tek aşamalı (yani bir şey çizmek için başka birşey yapmamız gerekmez, hemen başlarız) olmasıdır. Bilgisayar ile yapılan her iş için bilgisayarı açmamız, programın açılmasını beklememiz, program açıldıktan sonra ise araç seçmek ve çıkarken kaydetmek gibi işleri yapmamız gerekir. Her ne kadar bu işler uzun süreli (saatlerle ifade edilen) bir çalışmanın içinde nispeten kısa bir süre alsa da, tasarımın erken aşamaları gibi kısa süreli ve aralıklı çalışmalarda ,çalışmanın kendisinden bile fazla vakit alabilir.

Örneğin, tasarımın erken aşamalarında, hareket halinde ve yada bilgisayar başında zihinde canlanan bir fikri bilgisayara (kağıda) döken, daha sonra bilgisayarı(defterini) kapatıp bir saat sonra bir kafeterya da ekrandakilerin(kağıttakilerin) üzerinden düşünüp birşeyler çizen, sonrasında gün bitiminde eve döndüğünde de yatmadan önce çizdiklerine bakan bir tasarımcı için, her seferde kullandığı yazılımı/aleti tekrar çalıştırması bile ,o işi yapmasına engel olacak şekilde uzun bir vakit alır. Bilgisayardaki teknik çizim yazılımlarında hiç bir zaman eskizdeki gibi kolayca tek aşamada çizgi çizemezsiniz (Davis, 2007)

Tüm bunların dışında kağıt üzerinde eskizin, üzerine bir eşya koyarak onun kontürünü çizebilmemiz, altına başka bir çizim koyup üzerinden kopya edebilmemiz, kağıdı ikiye bölüp ilgili kısımları başkasına verebilmemiz gibi, eskizin fiziksel durumuyla ilgili avantajları da vardır. Bu nitelikleri herhangi bir bilgisayar yazılımının da aramak anlamsız/imkansız gibi görünse de, çizim yüzeyinin üzerine veya altına eşya koyup kopyasını çekmek, yarı saydam ve dokunmaya duyarlı ekranlarda olduğu gibi, bilgisayar ve malzeme teknolojisindeki gelişmeler ile bugün teknik olarak mümkün olabilmektedir (**Şekil 1.4**).



Şekil 2.4 Altındaki çizimi gösterebilen saydam tablet bilgisayar(konsept ürün tasarımı- Iris Transparent tablet project, Liu-Wei, Kai-Chi Ruei-Hong 2008)

2.2 Eskize Öykünen Öncül Yazılım ve Donanımlar

1960'lı yıllardan itibaren, bilgisayar kullanıcı arayüzlerinde klavye ile metin girme dışında çeşitli araçlar da kullanılmıştır. Bu araçlardan fare veya elektronik kalem gibi iki boyut üzerinde pozisyon bilgisi verebilen araçlar, farklı şekilde kullanıcısıyla ilişki kuran yazılımlara izin verdi (Kay, 1987). Bu işaretçilerin (fare, kalem vb.) kullanımı, düşünürken/çalışırken kullandığımız , kağıt-kalem ortamından alışık olduğumuz, melekeleri/pratikleri bilgisayar ortamında kullanılmasına yönelik birçok yazılım ve donanımlar geliştirilmesine olanak sağlamıştır. Çoğu yazılım pozisyon bilgisi giren bu

araçları grafik menülerin kullanımı veya program içinde bunların hareketinden dolayı oluşan izi kullanarak bir altlık üzerinde çizim yapabilmesi için kullanırken bazı geliştiriciler, bu özel durumun kendine has potansiyelinin¹ ne olabileceği hakkında çalışmalar yapmış, yeni program kullanma şekilleri/arayüzleri üzerine gitmiştir.

Tezin bu kısmında bahsi geçen kağıt-kalem ile eskiz ilişkisine benzer bir ortamı bilgisayarın da kabiliyetlerini katarak arayüzlerinde sağlamaya çalışan yazılımlar ve donanımlar incelenmekte ve çalışma şekilleri anlatılmaktadır. Tez çalışmasının amacı olan, insanın kağıt-kalem ile düşünme sürecinin önüne en az geçerek , bilgisayarın kabiliyetlerini de kullanmayı amaçlayan yazılım önerisi için , aşağıda anlatılan yazılım ve donanımların kullanım şekilleri ve yarattıkları çalışma ortamı önemli bir kaynak olmuştur.

2.2 Yazılım ve Donanım Örneği : SketchPad / 1963

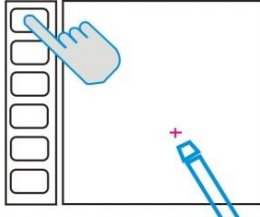
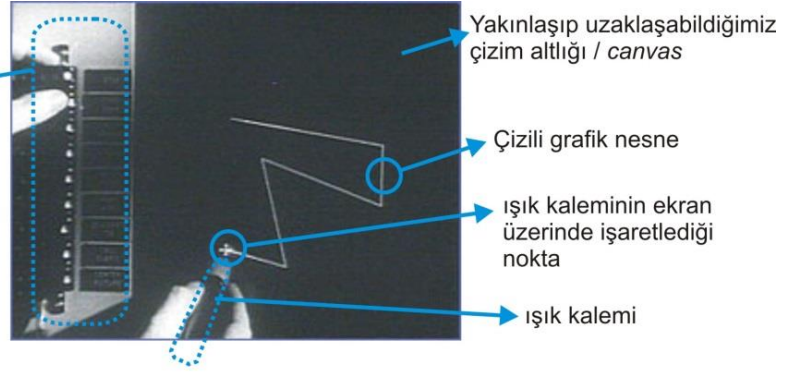
SketchPad 1963 yılında Ivan Sutherland tarafından geliştirilmiş bir yazılım ve yazılımın çalışabildiği donanıma verilen isimdir. SketchPad, kullanıcının bilgisayar ekranı üzerine bir ışık kalem ve komut düğmeleri yardımı ile geometrisi bilgisayar tarafından tanımlanmış şekiller çizabildiği, bu şekiller üzerinde sonradan işlemler yapabildiği ve birbirleriyle etkileştirebildiği bir yazılımdır (**Şekil 2.5**)

Ekrana çizilen geometrik nesneleri oluşturan kordinat noktaları, yarıçap, uzunluk vb. nitelikler yazılım içinde değiştirilebilir ve kopyalanabilir değişkenler olarak saklanır. Bu sayede şekli çizdikten sonra da üzerinde değişiklikler yapılabilir. Kopyalanan biçim, kopyalandığı biçimin tanımının aynısına sahip yeni ve bağımsız bir biçim olabilir ya da kopyalandığı asıl biçimin somutlanan örneği (Instance²) olabilir. Birbirlerinin somutlanan örneği olan biçimlerin birisinde yapılan değişiklik bir diğer nesneyi de etkiler. Mesela birbirlerine uç uca bağlı çizilmiş doğru parçalarının, birbirinin somutlanan örneği olarak oluşturulan yeni kopyaları, asıl şekilde yapılacak değişiklikten aynı şekilde etkilenir (**Şekil 2.6**). Çünkü geometrileri oluşturan öğeler arasındaki koordinat ilişkisi, nesnenin referans noktasına bağlı izafi bir ilişkidir.

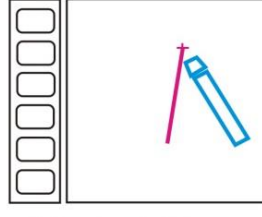
1 Burada bahsedilen potansiyel, bilgisayarın kendisine daha önceden detaylı anlatılan birçok şeyi aynı anda uygulayabilme gücünün, insanın kağıt kalemle iş yaparken de kullandığı bir yüzey üzerinde konum ve izlerden oluşan ortam ile beraber düşünülmesidir.

2 Instance : Somutlanan örnek. Soyut bir bilgi topluluğu ilişkisinin, değişkenlerinin değişmesiyle gerçekleşen bir örneği, yazılım terimi.

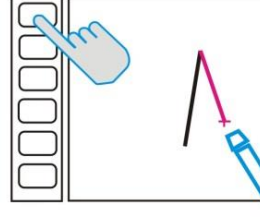
Çizim fonksiyonlarını (çizgi, , daire vb.) ve çizim yardımcı araçları ve kısıtlamaları (kopyalama , bir eksene sabitleme, nokta yakalama vb.) barındıran tuşlar. Bu tuşlara basarak belirlenen nesneler ve fonksiyonlar ekrana tutulan ışık kalemiyle çizilir.



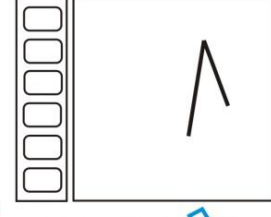
Işık kalemi ekrandaki bir noktaya tutulup çizgi tuşuna basıldığında o nokta çizginin başlangıç noktası olur ..



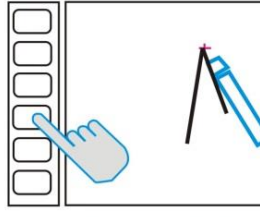
.. Kalem hareketiyle eş zamanlı olarak, kalem ne hedefe tutulursa çizginin son noktası orası olacak şekilde bir çizgi oluşur



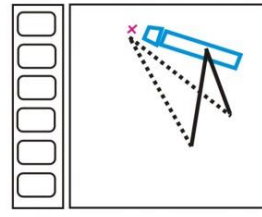
.. İlgili tuşa tekrar basılırsa, basılma anındaki nokta 1. Çizginin bitiş noktası ve yeni çizginin başlangıç noktası olur



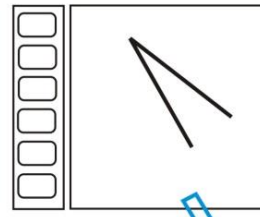
Işık kalemi hızlı şekilde ekran dışında bir noktaya tutulduğunda, işlem sonlanır



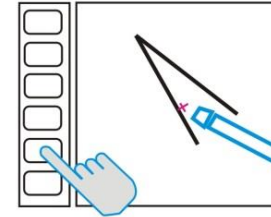
Işık kalemi ekrandaki bir noktaya tutulup çizgi tuşuna basıldığında o nokta çizginin başlangıç noktası olur ..



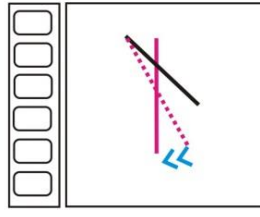
Işık kalemi ekrandaki bir noktaya tutulup çizgi tuşuna basıldığında o nokta çizginin başlangıç noktası olur ..



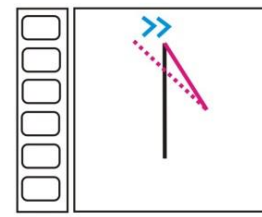
Işık kalemi hızlı şekilde ekran dışında bir noktaya tutulduğunda, işlem sonlanır



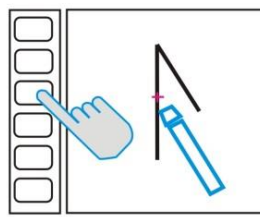
Işık kalemi bir çizginin üzerine tutulurken, düşey yatay eksenle sınırlandırma tuşuna basıldığında..



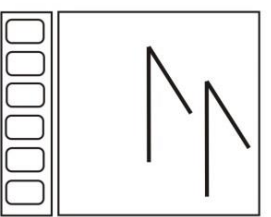
İşaretli doğru parçası işaretli noktası merkez kabul ederek düşeye hizalı hale getirilir ve..



.. Hizalanan doğrunun komşusu doğrunun, kesişim noktası da, dönen doğruya uyacak şekilde kayar.



Bir çizginin üzerine gelip kopyalama tuşuna bastığımızda ...

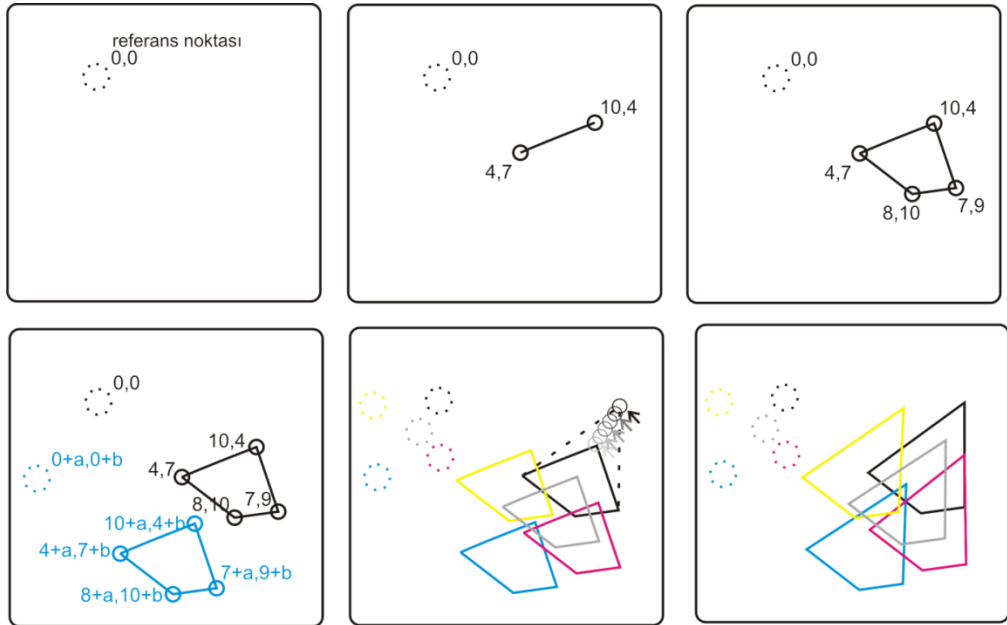


.. Nesneyi kalemle istediğimiz pozisyona yetirip, hızlıca ışık kalemini ekran dışında bir noktaya tuttuğumuzda kopyama biter

Şekil 2.5 : Sketchpad ile yapılabilen temel çizim fonksiyonları ve kullanıcı tarafından nasıl kullanıldığını anlatan tablo (Science Reporter, 1964)

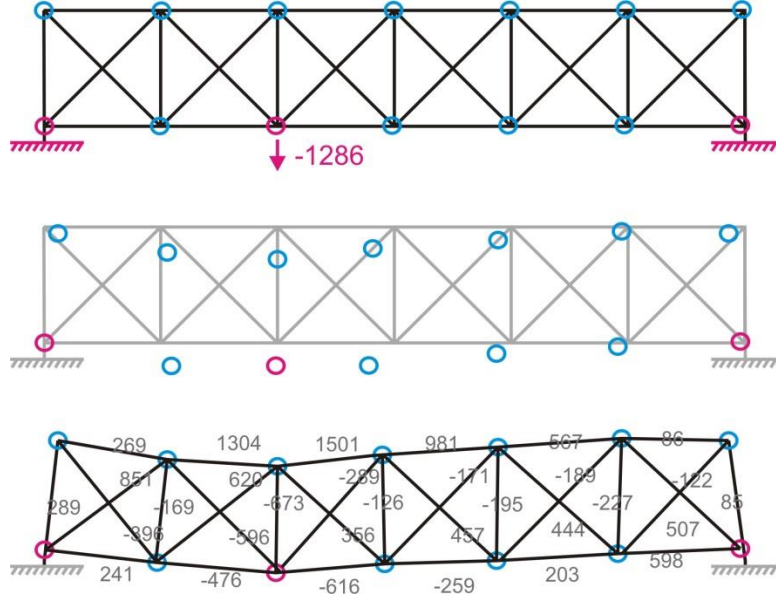
Kordinat düzlemi üzerinde yer değişikliğinde, şekil yeni referans noktasına göre tekrar hesaplanır. Bu hem birbirlerinin tekrarı olan şekiller üzerinde çalışırken kullanıcıya yapılacak değişiklikler için çok büyük bir kolaylık sağlar, hem de bilgisayarı birbirinin aynısı olan hesaplamaları yapmaktan kurtarır . Bu yapı daha sonraları nesne tabanlı programlanmanın da temelini oluşturacaktır (Key, 1983)

Sutherland, Sketchpad'ın dört farklı duruma sahip işlerde kullanımının süreci kolaylaştırdığının açıkça görüldüğünü belirtmiştir. Bu durumlar: var olan çizimler üzerinde küçük değişiklikler yapılabilmesi, grafik olarak ifade edilen bilimsel/mühendisliksel operasyonların özümsebilmesi, grafik nesneler arasındaki topolojik özelliklerin saklanabilmesi ve çok fazla tekrara sahip çizimlerin çizilebilmesidir (Sutherland, 1963). Şekil 2.7'de grafik olarak ifade edilen mühendislikle ilgili operasyonlara örnek olarak, eksenel yük altında statik nitelikleri incelenen bir kafes kiriş hesabı örnek verilebilir. Yük vektörünün, mafsalların ve taşıyıcı kafesin SketchPad'de çizildiği şemada, kuvvet vektörünün değdiği nokta, vektörün kuvvetine ve yönüne göre bir miktar yer değiştirir (Şekil 2.7). Ona bağlı tüm kafeslerde yer değiştirme meydana gelir. Fakat mafsal noktaları da sabit kalıp bu yer değiştirmeye karşı bir durum oluşturur.



Şekil 2.6 : SketchPad ile birbirinin somutlanan örneği olan nesnelerin birinde yapılan değişikliğin bir diğerini etkilemesi

SketchPad bütün birleşim noktaları için bu iki çekim noktasına göre yakınsama¹ işlemi yapar ve işlem sonucunda sistem dengeye gelir. Kafes kiriş noktalarını birbirine bağlayan çubuk elemanların, ilk çizildiğindeki uzunluğuyla son uzunluğu arasındaki fark çubukların yanına yazılır. Sonuç olarak çıkan bilgiler çubuk elemanların üzerindeki gerilimi gösteren bir şema olur.

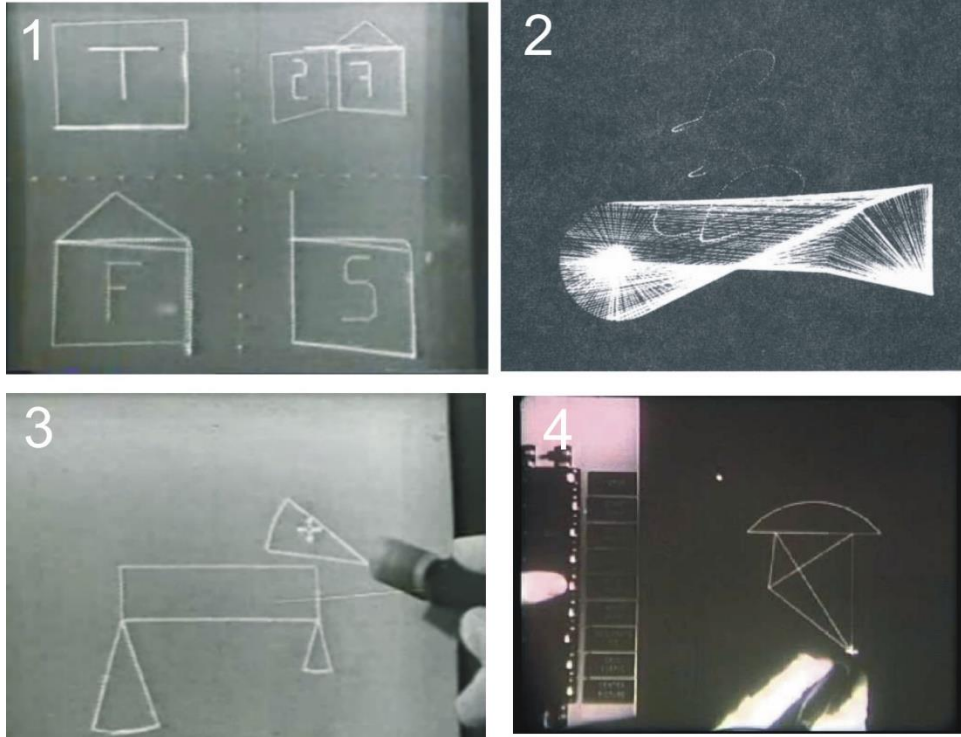


Şekil 2.7 : SketchPad ile kafes kiriş kuvvet analizi. Yük vektörünün değdiği yerdeki geometrideki yer değiştirmenin zincirleme olarak bağlı olduğu geometrileri de değiştirmesi sonucu oluşan uzama kısaltmaların, elemanların yanına yazılması.

SketchPad bilgisayarda özel bir işin yapılması amacıyla tasarlanmış bir yazılım değildir ve kurgusunun imkan verebileceği herhangi bir iş için kullanılabilir. Ivan Sutherland, Sketchpad’i anlattığı makalesinin başlangıcında, SketchPad’i kullanım amacı üzerinden tanımlamak yerine, çizme eylemi üzerinden bilgisayar ve insan arasında iletişim sağlayan bir ortam olarak tarif eder (Sutherland, 1968). SketchPad ile desenlerinin ve ölçeğinin değiştirilip sonuçlarının eş zamanlı ekranda görülebilebildiği doku çalışması, bir boyutta çizilenin diğer boyutlarda da eş zamanlı yazılım tarafından çizildiği üç boyutlu çizim, çizgisel animasyonlar, taşıyıcı eleman kuvvet şemaları, devre şemaları gibi birçok çeşitli iş yapılabilir (Şekil 2.8).

¹ Yakınsama : Sutherland’in SketchPad’i anlattığı makalede “Relaxation” olarak geçen , biçimleri hizalamaya, birbirlerine tutunmasına, yanaşmasına yarayan kısıtlamaların (constraints) yöntemi.

SketchPad, Engelbart'ın tasarladığı On-line Sistem'e (NLS computer system), Xerox PARC'da tasarlanan SmallTalk gibi nesne tabanlı programa dillerine ve Alto bilgisayarların grafik tabanlı arayüzlerine, Autocad gibi vektörel çizim programlarına kadar birçok çalışmaya ilham kaynağı olmuştur (Url-1). Bütün biçimlerin bir grafik nesne olarak saklanması ve nesnelerin altındaki çeşitli niteliklerin (uzunluk, çizgi tipi gibi) değiştirilebilmesi ve etkileşebilmesi, SketchPad'in kalem-kağıda benzer grafik arayüzü ile beraber , tez çalışmasının amacı olan yazılım önerisinin tasarım ilkeleri için önemli bir ilham kaynağı oluşturmıştır.

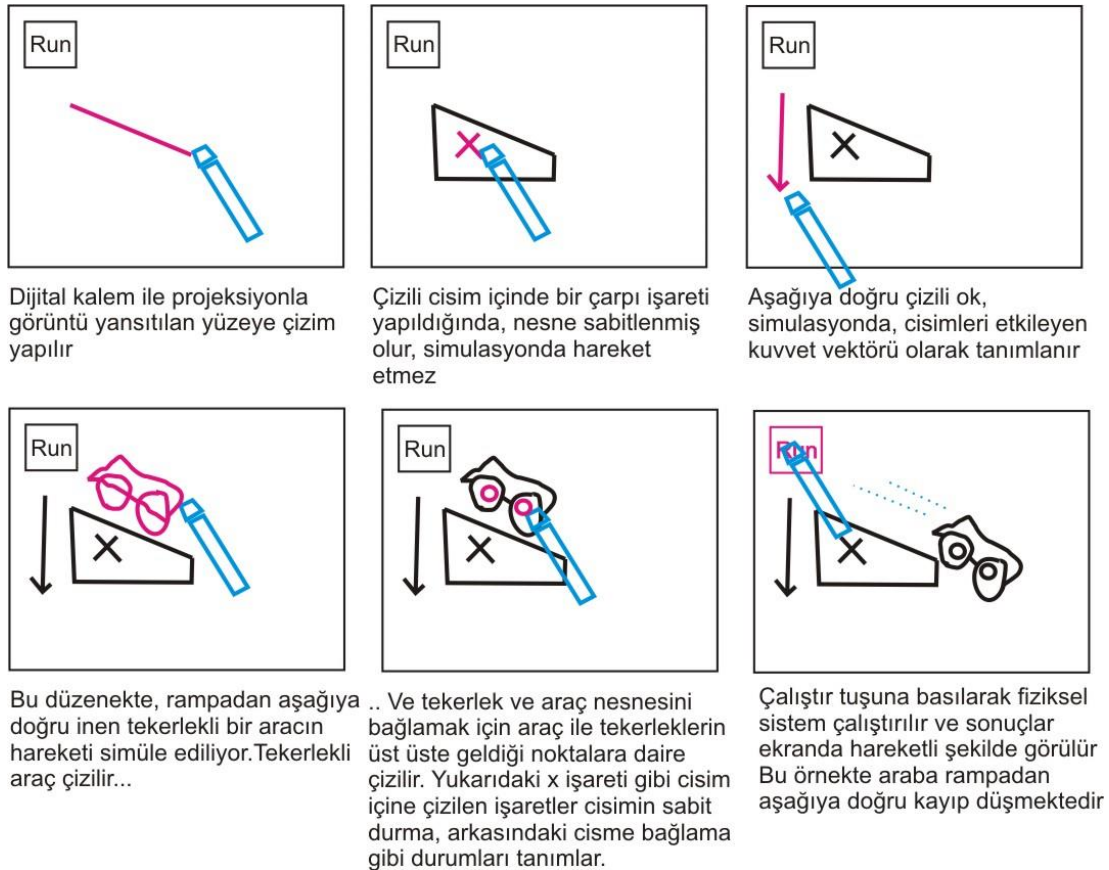


Şekil 2.8 : SketchPad'in kullanım alanlarından örnekler: 1- Üç yönden ve perspektiften görülen üç boyutlu çizim, 2- Çizginin zamanla şekil değiştirmesiyle oluşan animasyon, 3- *Instance* nesneler, 4-Biçimlerin sonradan manipule edilmesi

2.3 Yazılım Örneği : Assist / 2000

Assist, elektronik kalem ile duvar veya beyaz tahta gibi bir yüzey üzerinde çizilmiş fiziksel deney düzeneklerinin simülasyonlarının gerçekleştirilebildiği, M.I.T. Design Rationale grubu tarafından tasarlanmış bir sistemdir. Projeksiyonla görüntünün yansıtıldığı yüzeye elektronik kalemlle çizim yapılır ve çizilen izler, yazılım içinde önceden belirli kütle, kuvvet vektörü, mesnet vb. nesnelere benzetilerek bir fiziksel simülasyon düzeneği oluşturur.

Çalıştır tuşuna basıldığında bu fiziksel düzenek simülasyonu çalışmaya başlar ve ekrana çizilen nesneler, bu simülasyona göre hareket ederler ve tepki verirler. Yazılımı geliştirenler, bir yüzey (bu yüzey kağıt veya beyaz tahta olabilir) üzerinde mekanik sistem tasarlayan mühendislerin, kurdukları düzenek içindeki etkileşimi ve sonuçlarını, gerçek bir mekanik sistem gibi gözlemleyebileceği, yani çizdikleri şeyin, yine üzerine çizdikleri ortamda, gerçekleştiği halini görebileceği bir platform hazırlamayı hedeflemişlerdir (Davis ve Alvarado, 2001). Projeyi geliştirenler, ekrana el ile çizilen nesnelerin program tarafından ne olduğunun tanınabilmesi üzerine çalışmışlardır. Çizilen nesnelerin nasıl algılandığına örnek vermek gerekirse ; kapalı biçimler nesne gibi algılanıp, alanlarıyla doğru orantılı bir kütleye sahip olur, ok işareti ve yanına bir rakam yazıldığında nesneye o rakam kadar kuvvet uygulanır, nesne, ortasına çarpı işareti konulduğunda kuvvetlerden etkilenmez veya ucuna yaya benzer birşey çizildiğinde o çizilen nesne yaya takılıymış gibi hareket eder (Şekil 2.9).



Şekil 2.9 Assist programında çizilmiş basit bir fiziksel deney ortamı , pembe renkli ifadeler kuvvet yönü, mesnet ve sabit öğelerin işaretlenmesi için kullanılmış / ekran çıktısı

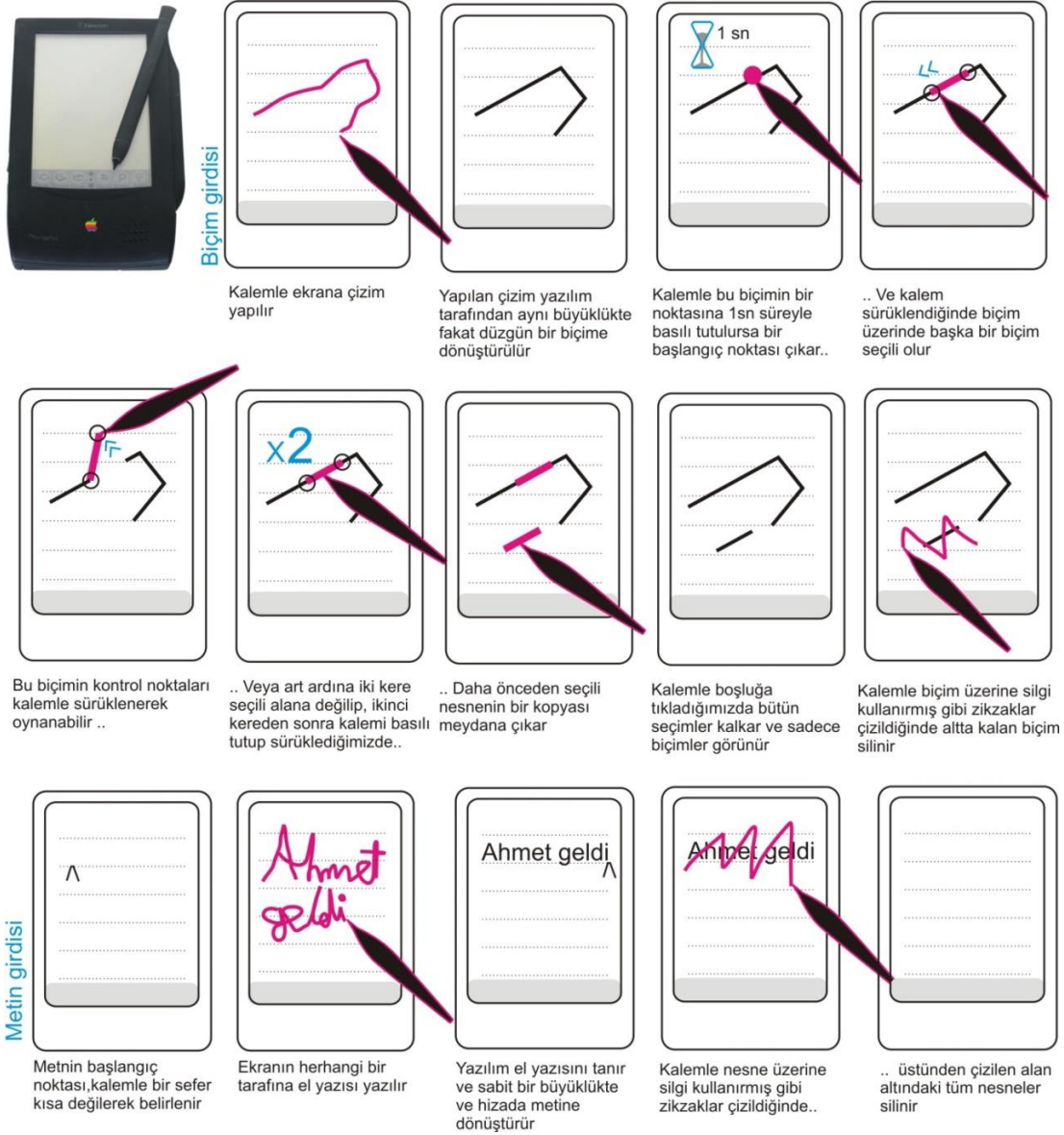
Bu kullanıcının ne söylemek istediğini tahmin eden akıllı kullanıcı arayüzü, başka hiçbir tuşa veya aşamaya ihtiyaç olmadan farklı niteliklerdeki girdiyi tek bir yöntem ile (sadece çizerek) yorumlar, kullanıcı farklı bir girdi girmek için ilk önce bir menüden girmek istediği girdi tipini belirlemek gibi bir ihtiyaç duymaz.

Yazılımın arayüzü ve kullanılma şekli, tasarımın ön aşamalarında kullanıcının işini çok kolaylaştırabileceği gibi, karmaşık problemlerde ve net sonuçlar alınması gereken durumlarda işe yaramayabilir (Davis ve Alvarado, 2000). Kullanıcının yazılımı kullanırken herhangi bir çoktan seçmeli menü kullanmaması ve kalemle çizmek dışında olabildiğince az arayüzle muhatap olması, bu çalışma için önemli bir örnek olmuştur..

2.4 Yazılım ve Donanım Örneği : Newton Platformu / 1993

Newton, bilgisayarın kullanıcılarının klavye ve fare gibi arayüz araçları dışında, kağıt kalem metaforuyla kullanabilecekleri bir bilgisayar platformudur. Apple firması bu platform için basınca duyarlı bir ekrana sahip, pille çalışan taşınabilen not defteri büyüklüğünde bir donanım ve yazılım imal etmiştir. Bu platformda üretilen bilgisayarlar üzerilerine çizilen grafik ifadeleri yorumlayarak metin veya geometrik biçime çevirme becerisine sahiptir. Bahsedilen metin, grafik ifade ve hareketler (*gestures*) Newton içindeki çeşitli yazılımlarda bilgi girme yöntemi olarak kullanılır (Apple, 1996).

Yazılım ve donanımın çalışması basit şekilde (anlatmak gerekirse); basınca duyarlı ekranın üzerinde kullanılan kalemle yapılan girdinin tipi çoktan seçmeli bir menüden seçilip (yazı veya serbest biçim olup olmaması gibi) daha sonrasında ekranın istenilen noktasına istenilen çizim yapılır. Yazılım da yapılan çizimi kendi belleğinde olan bir nesneye (harf veya üçgen gibi) benzeterek çizmek istediğimiz şeyi bilgisayarın kendinin de anlayabileceği/işleyebileceği/değiştirebileceği/kopyalayabileceği şekilde kaydeder(**Şekil 2.10**). Bu (üstünde yazı yazılabilen) ekranın üzerine yazılan yazı ve çizilenlerin, kağıt üzerine yazılan ve çizilenlerden farkı, daha sonra bu bilgilerin bilgisayara işletilebilinmesidir. Bir bilgisayar içine yazılan kelimeler daha sonra aratılabilir ve yine bilgisayarda çizilen bir dairenin büyüklüğü değiştirilip yanına birkaç kopyasını konabilir, fakat bunlar not defterinde yapılamaz.



Şekil 2.10 Newton işletim sisteminde, notepad yazılımına kalemle veri girişi

Buna karşın bilgisayarda yazı yazma ve çizim yapma işi defterin üzerine yazmaya göre çok basamaklı ve vakit alan bir iştir. Newton platformundaki bilgisayarlar yazı ve biçim girdisi için defter kaleme çok benzer tek basamaklı ve kolay bir etkileşme yöntemi kullanarak bilgisayarların bu dezavantajı yenmeye çalışmıştır.

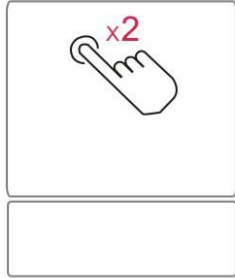
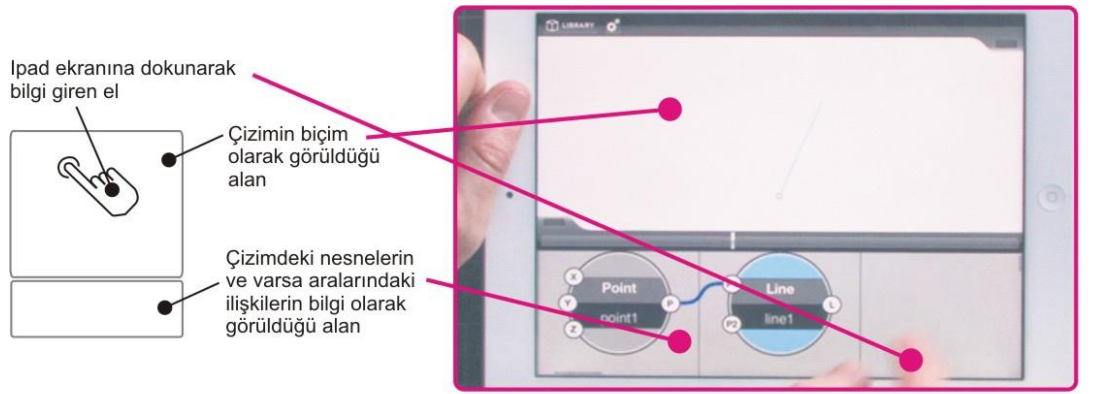
2.5 Yazılım Örneği : Dynamic Geometry Sketching / 2012

Daniel Belcher'ın daha tam olarak bitirilmemiş olan bu yazılımı, Ipad'lerde kullanılmak üzere, kullanıcının birbirleriyle ilişkili 2 ve 3 boyutlu şekiller (burada bahsi geçen ilişki, şekillerin arasında, şekillerin yer, büyüklük uzunluk ve benzeri niteliklerindeki değişiminin, bir diğer şeklin benzer niteliklerini değiştirmesidir) çizmesine ve bunların nitelikleriyle çizildikten sonra da oynanmasına izin veren bir yazılımdır (Url 2). Yazılımın biçimlerin ifade edildiği ve biçimlerin niteliklerinin ifade edildiği iki ekranı vardır. Birinci ekran biçimlerin uzaydaki görünüşlerini gösterir, ikinci ekran ise biçimleri, kendisi ve niteliklerinin başka bir niteliğe bağlanabilen bir nokta ile ifade edildiği bir arayüzü (*node based operator*) gösterir.

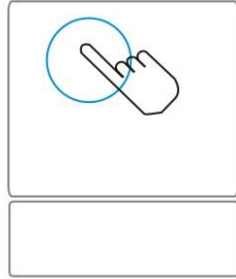
Yazılımda iki ekrandan istenilen üzerinden yeni bir grafik nesne eklenebilir. Örnek olarak vermek gerekirse üstteki ekrana bir defa dokunmamızda ekranda bir nokta belirir ve o noktadan bir yanına elimizle bir çizgi çizdiğimizde, çizdiğimiz biçim o noktaya bağlı düzgün bir doğru oluşturur. Aynı anda da alttaki ekranda yukarıdaki ekrandaki biçimlerin nitelikleri ve ilişkileri eşzamanlı olarak belirir. **Şekil 2.11** 'de ki diyagramda programın kullanılışı anlatılmıştır. Daha sonrasında da kullanıcı isterse ekranda cisimlerin yerlerini değiştirerek, isterse aşağıdaki nesne ve parametrelerle oynayarak, ekrandaki geometrileri manipule eder . Alttaki nesne menüsündeki parametreler aynı zamanda başka nesnelere de bağlanabileceğinden, bir nesnedeki biçim veya konumla ilgili değişim , diğer bir nesnenin de biçim veya konumunu etkileyebildiği karmaşık ilişkiler kurulabilir.

2.6 Yazılım ve Donanım Örneği : Glassified Ruler / 2013

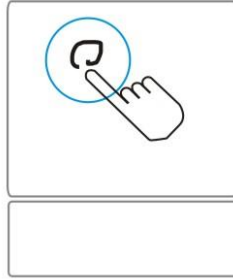
MIT Media Lab Fluid Interfaces araştırma (Anirudh Sharma, Lirong Liu ve Pattie Maes) grubunda yapılmış , saydam bir cetvelin bir kısmına yine saydam bir ekran, kamera ve küçük bir bilgisayar yerleştirerek yapılmış bir elektronik cetveldir (**Şekil 3.8**).Bu cetvele bağlı dijital kalem ile kağıt üzerine çizdiğimiz biçimler cetvel üzerindeki bilgisayara iletilir ve bu biçimler üzerinde önceden tanımlı çeşitli işlemler uygulanır ve sonuçlar saydam ekran üzerine yazdırılır (Url 3). Üzerinde, daha önceden çizdiğimiz şekil beliren saydam cetveli kağıt üzerinde çizdiğimiz şekil ile üst üste getirdiğimizde, ekrandaki bilgiler anlamlanır. Bu cetvele, kağıt üzerine çizdiğimiz bir şeklin alan ve uzunluklarını hesaplatmak, kağıt üzerine çizilen çizgilerle ilişik animasyonlar üretmek gibi birçok işlev verilebilir (**Şekil 3.8**) .



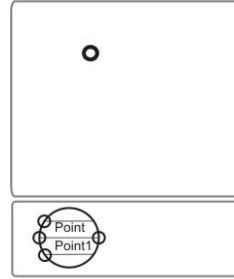
Ekranı iki kez değerek o konuma bir nesne konur



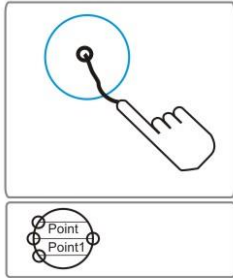
Hemen ardından büyük bir daire belirir ve içine çizmek isteyeceğimiz biçime karşılık gelen bir hareket çizilir



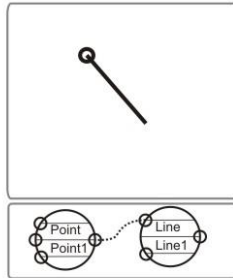
Buraya bir nokta koymak istiyorsak bu hareket küçük bir daire şeklindedir



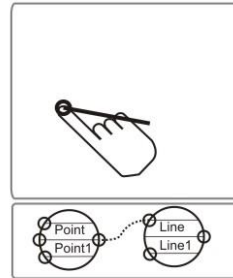
Üst pencerede bir nokta ve alt pencerede o noktaya ait bilgiler belirir



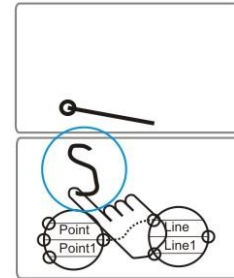
Bu noktaya bağlı bir doğru çizmek için nokta için yaptığımız eylemlerin aynısı yapılır, sadece daire şeklinde bir hareket değil doğruya benzer bir hareket yapılır



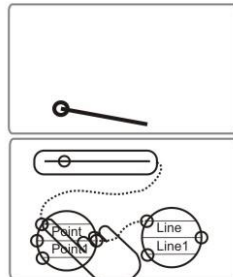
Üst pencerede bir doğru ve alt pencerede nokta ile ilişkisi gösterilmiş doğru nesnesine ait ifade belirir



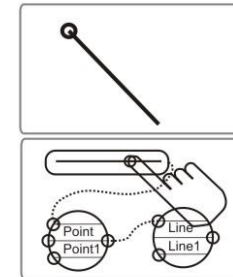
Üst ekrandaki doğrunun başlangıç noktasının yeri hem parmağımızla sürükleyerek değişebilir hemde..



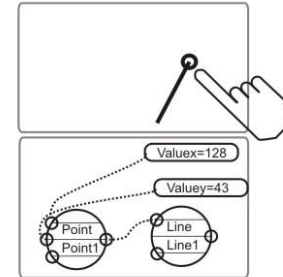
.. üst ekrana nesne eklemeye benzer şekilde alt ekrana iki kere tıklayıp konulacak nesneye ait hareketi çizerek..



.. Bir slider nesnesi oluşturup, noktanın x eksenine bağlayarak..

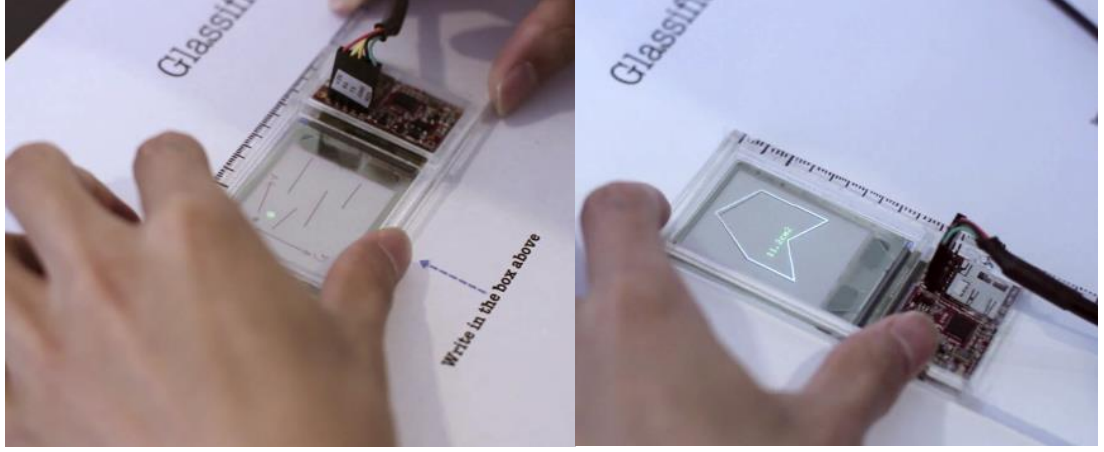
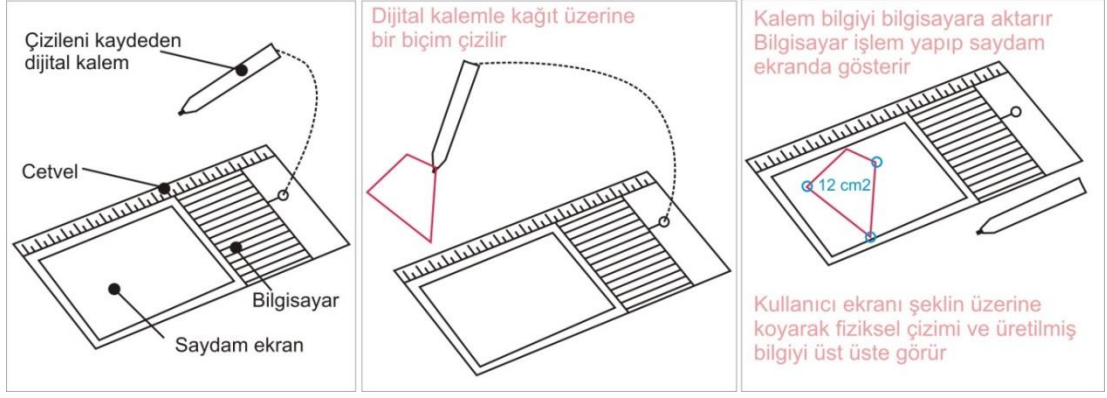


.. Aşağıdaki ekrandan yukarıdaki çizimi kontrol edebiliriz



Aynı şekilde yukarı pencerede yapılan değişikliğin sonucunu aşağıda görebiliriz

Şekil 2.11 Sketching Dynamic Geometry yazılımının örnek kullanımı



Şekil 2.12 Glassified Ruler çalışma şeklini anlatır şema ve uygulama örnekleri. İlk fotoğrafta hareket vektörü ve engelleri çizilmiş şekil içinde bir topun fiziksel hareketinin modellemesi, ikinci resimde kağıt üzerine çizilen şeklin alan hesabı uygulaması gösterilmektedir.

3. BİR MODEL ÖNERİSİ

3.1 Eskizin Kullanım Senaryolarının, Bilgisayardaki Mevcut Muadilleri ve Yorumlanması

Bu bölümde, kâğıt kalem ile yapılan eskizin bir durumu metin veya şekil ile kaydetme, yeni bir durum/ürün üretme, farklı şekillerde bilgiyi yanyana tasfîr etme gibi çeşitli amaçlarda kullanımı hakkında örnekler verilecektir. Bunu örnekler üzerinde durulurken, bu örneklerdeki işin bugün bilgisayarda nasıl yapılabildiği anlatılacak ve karşılaştırılacaktır. Bu işlerin iki ortamdaki yapılış aşamalarının irdelenmesi sonucu, bilgisayarların da üstlenebileceği tekrarlı/kurallı işler kurgulanarak, bilgisayar ortamında eskizin potansiyelleri ortaya çıkarılmaya çalışılmaktadır. İlk iki alt başlıkta yöntem olarak kalem ile kâğıt üzerinde biçim ve metin ile yapılan tasvirle oluşturulan ortamda yapılan iş/anlatım ile ,yine bu iş/anlatımların yapılabileceği mevcut bilgisayar yazılımları, tasvirin yapılma zorluğu, doğruluğu ve anlaşılır olması gibi farklılaşması göze çarpan nitelikler açısından karşılaştırılacak ,ve iki ortamında kuvvetli olduğu nitelikleri üzerinden yeni melez yazılım parçası önerileri anlatılacaktır.Üçüncü alt başlıkta ise Vannevar Bush'un 'As we may think' (1945) makalesinde bahsettiği Memex bir kavram olarak ele alınıp, internet ve eskiz kavramlarıyla beraber tekrar düşünülecek, olabilecek uygulamalardan bahsedilecektir. Tüm bu tespit ve öneriler bir sonraki bölümde anlatılacak yazılım için girdiler içermektedir.

Alt başlıklarda anlatılan çalışma senaryolarında, bahsi geçen işin eskiz ve bilgisayarda yapılma metoduna karşı önerilen yeni metotlar için düşünülmüş bir donanım üzerinde çalışan yazılım parçalarından söz edilmektedir. Bu yazılım parçalarının üstünde çalışacağı varsayılan donanım, tablet bilgisayar formunda, hem kalem ile hem parmak ile ayrı pozisyon verisi girilebilen, bunun dışında kamera vb. sensör/donanımlara sahip, orta büyüklükte bir not defteri boyutlarında bir bilgisayardır. Tezin sonuç bölümünde ise yazılımın kullanımını iyileştirebilecek fakat şu anda tablet bilgisayarlarda genel olarak kullanılmayan, teknolojilerinden ve faydalarından da bahsedilecektir.

3.1.1 Sonuca Denemelerle Yakınsayarak Ulaşıldığı Eskiz Çalışmaları

Erken aşamadaki tasarım ve denemeler için kağıt kalem ile eskiz yaparak çalışma, sıklıkla başvurulan bir yöntemdir. Makalenin 2.1.2'nci bölümünde de anlatılan çizilen şeklin çizim esnasında yavaş yavaş oluşması, çizenin kağıt üzerindeki tamalanmamış çizimden aldığı geri bildirimlerin önemli olması gibi nedenler, kağıt üzerinde defalarca üst üste çizerek yeni bir nesne biçimlendirmeyi, erken tasarımda kullanılan bir yöntem kılmaştır. Aşağıda eskizin bahsi geçen kullanımına ilişkin uygulamaya örnek olarak iki farklı senaryoda, kağıt kalem ile ve bilgisayarda iki boyutlu CAD programıyla (Autocad) ortaya çıkan iki farklı denemenin sonuç ürünleri sunulup, aşamalandırılıp, süreç ve sonuç ürünün bazı temel nitelikleri bakımından karşılaştırılmaktadır. Daha sonrasında ise bu iki farklı yöntemin olumlu görülen niteliklerini kapsayabilecek alternatif üçüncü bir yöntem önerilecektir. Bir tablet bilgisayar üzerinde çalışacağı var sayılan bu üçüncü yöntem tez çalışmasının amacı olan, eskiz yöntemi ile bilgisayarın kabiliyetlerinin birbirini örtmeden beraber kullanımı çerçevesinde düşünülmüştür.

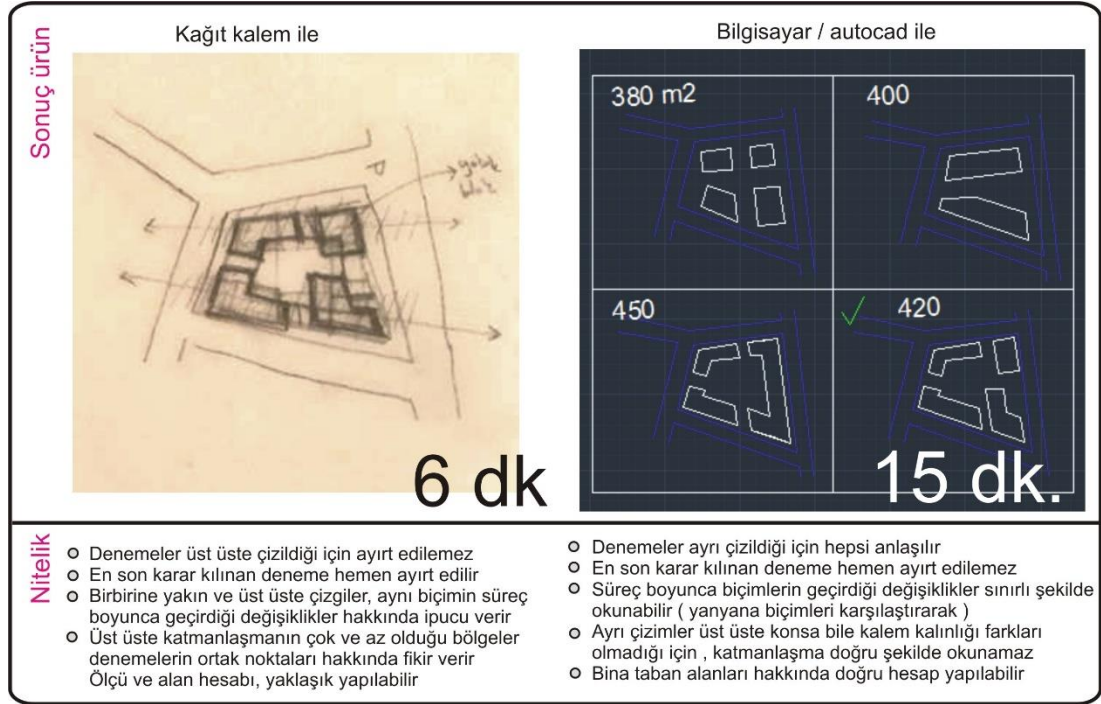
3.1.1.1 Üst üste yapılan tekrarlı çizim ile tasarım senaryosu

Bu senaryoda kullanıcı, iki konut bloğunun vaziyet planında yerleşimine karar vermektedir. Birinci denemesinde Autocad programında, mevcut vaziyet planı üzerine dört farklı öneri çizmekte, ikinci denemesinde ise vaziyet planı çıktısı üzerine yarı saydam eskiz kağıdı koyarak eskiz yoluyla dört farklı deneme yapmaktadır. Bu iki deneme sonucu çıkan ürünler fotoğraflanmış, harcanan zaman ölçülmüş ve iki yöntemin ve sonuç ürünün farklı niteliklerinin karşılaştırılması tabloalanmıştır. **Şekil 3.1**'de karşılaştırılan iki yöntemin birbirinden daha kolay olmasını sağlayan ve sonuç ürünün farklı okumalara izin vermesi hususunda öne çıkan nitelikleri:

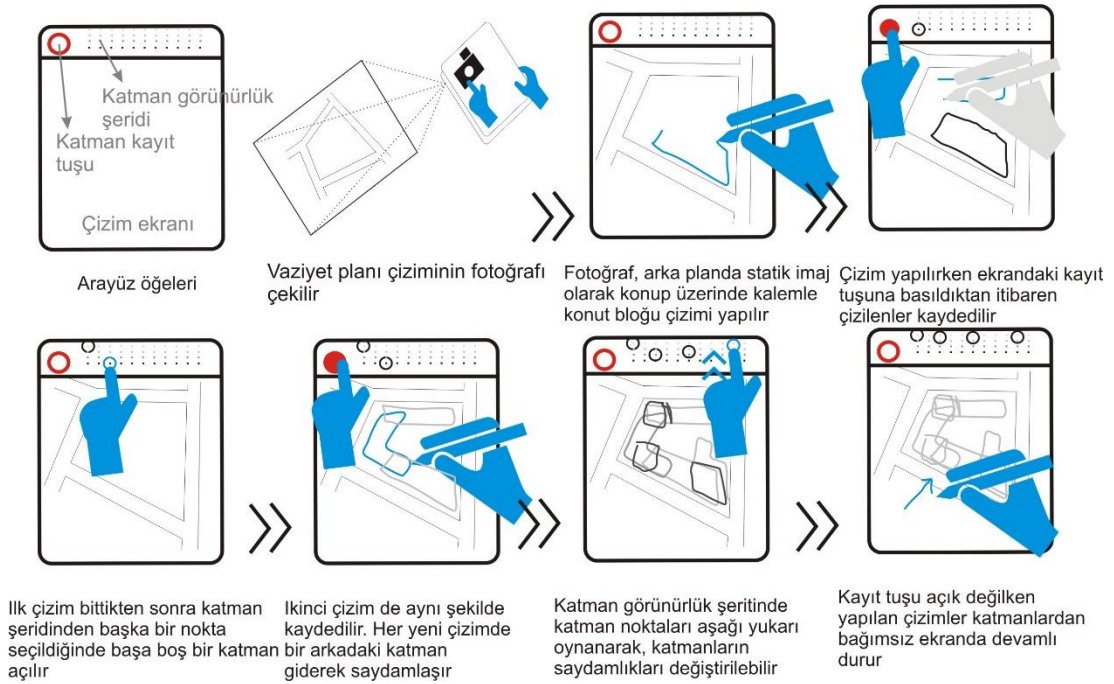
- Bilgisayarın öne çıkan niteliği olarak; çalışmanın her aşamasının ayrı ayrı görülebilmesi ve çizim üzerinde ölçüm ve hesap yapılabilmesi.
- Eskizin öne çıkan niteliği olarak; daha hızlı ürün elde edilmesi, denemelerin üst üste olması nedeniyle çalışma aşamalarının ve istikametinin daha sonra okunabilmesi

Yukarıda bahsi geçen niteliklerin beraber sağlanması amaçlayan üçüncü yolda, tablet bilgisayarda fiziksel eskiz gibi bir boşluğa serbest el ile çizim yapılabilmekte fakat eskiz yapılırken kullanıcı isterse ekranda bir alana parmağı ile değerek o anda çizdiği biçimi diğerlerinden kopuk bir biçim olarak tanımlayabilmektedir.

Bu önerilen üçüncü yolun kullanım senaryosunda, kullanıcı tablet bilgisayar ile imajın fotoğrafını çekip üzerinden kalem ile vaziyet planını çizmekte, daha sonrasında da üzerinde eskiz ile serbest denemeler yapmaktadır.



Şekil 3.1 Vaziyet planı üzerinde konut bloğu tasarımı / Fiziksel eskiz ve CAD programındaki çalışmaların karşılaştırılması



Şekil 3.2 Öneri Yazılımla Vaziyet planı üzerinde konut bloğu tasarımı

Fakat her deneme başlangıcında parmağı ile ekranda bir alana dokunacak, ve bilgisayarda bunu anladığında bu eskizi ayrı bir katmanda kaydedip kullanıcı istediğinde geçmişte yaptığı eskizleri ister teker teker, ister üst üste çağırabilecektir. Kullanıcı daha sonra bu eskizlerden beğendiğini ekranda bir alana dokunarak seçecek, bu seçim sonucunda bu denemesi diğer çizimlere göre daha baskın renk ile ifade edilecektir (**Şekil 3.2**). Böylece kullanıcı autocad çizimine göre daha az aşamalı ve fiziksel eskize benzer bir yolla ürün çıkartacak, ve bu ürünün çşitli aşamalarını sanki bilgisayarda çizmiş gibi istediğinde ayrı ayrı görebilecektir.

3.1.1.2 Karşılıklı iki nesnenin çifter denemelerle tasarımı senaryosu

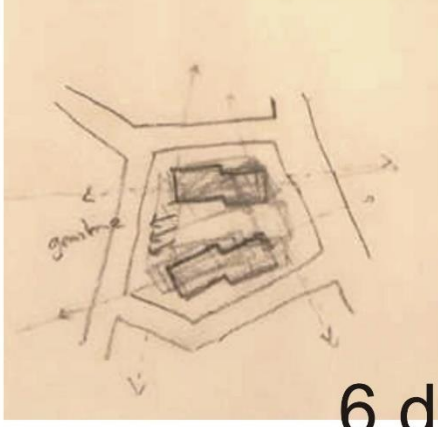
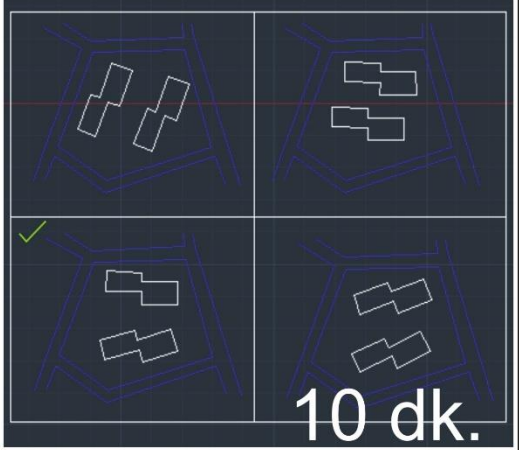
Bu senaryoda kullanıcı biçimi belli iki konut bloğunun vaziyet planında yerleşimini çalışmaktadır. Birinci denemesinde Autocad programında biçimi belli blokların yerleri ve yönlenmeleriyle oynayarak dört farklı yerleşim denemesi yapmakta, ikinci denemesinde ise vaziyet planı çıktısı üzerine saydam eskiz kağıdı koyarak üzerinde kalemle denemeler yapmaktadır.

Bu iki deneme sonucu çıkan ürünler fotoğraflanmış, harcanan zaman ölçülmüş ve iki yöntemin ve sonuç ürünün farklı niteliklerinin karşılaştırılması tablolanmıştır. **Şekil 3.3**'de karşılaştırılan iki yöntemin birbirinden daha kolay olmasını sağlayan ve sonuç ürünün farklı okumalara izin vermesi hususunda öne çıkan nitelikleri:

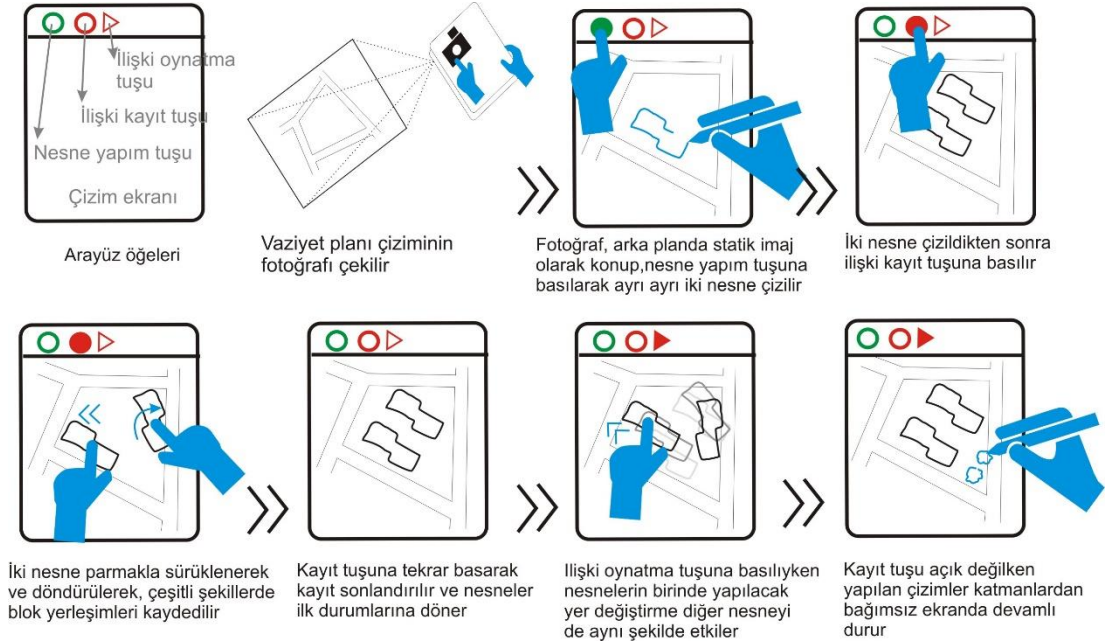
- Bilgisayarın öne çıkan niteliği olarak; çalışmanın her aşamasının ayrı ayrı görülebilmesi ve çizim üzerinde ölçüm ve hesap yapılabilmesi (bir önceki örnekle aynı sonuçlar, fakat burada yapı biçimi bir defaya mahsus çizilip, bunu kopyalarıyla oluşturulduğu için çok daha kısa zamanda ürün çıkıyor).
- Eskizin öne çıkan niteliği olarak; daha hızlı ürün elde edilmesi, denemelerin üst üste olması nedeniyle çalışma aşamalarının ve istikametinin daha sonra okunabilmesi (bir önceki örnekle aynı sonuçlar, fakat burada denenen biçimlerin üst üste beraber okunurluğu daha az).

Yukarıda bahsi geçen niteliklerin beraber sağlanması amaçlayan üçüncü yolda, tablet bilgisayarda fiziksel eskiz gibi bir boşluğa serbest el ile çizim yapılabilmekte fakat eskiz yapılırken kullanıcı isterse ekranda bir alana parmağı ile değerek çizdiği iki ayrı yapı biçimini aynı anda hareket ettirebilmekte ve yaptığı bu eşli hareketi daha sonar bilgisayara tekrarlatabilmektedir.

Üçüncü yol olarak önerilen yazılımın kullanım senaryosunda kullanıcı tablet bilgisayar ile imajın fotoğrafını çekip üzerinden kalem ile vaziyet planını ve konut bloklarını çizecektir.

	Kağıt kalem ile	Bilgisayar / autocad ile
Sonuç ürün		
	6 dk	10 dk.
Nitelik	- Denemeler üst üste çizildiği için ayırt edilemez - En son karar kılınan deneme hemen ayırt edilir - Yapıların yollar ve birbirleri arasındaki mesafeler ancak yaklaşık olarak tahmin edilebilir - Biçimlerde bir evrimleşme değil yalnız yer değiştirme olması yüzünden, üst üste katmanlaşmanın vereceği bilgi potansiyeli kullanılmamış olur.	- Denemeler ayrı çizildiği için hepsi anlaşılır - En son karar kılınan deneme hemen ayırt edilemez - Yapıların yollar ve birbirleri arasındaki mesafeler tam olarak ölçülebilir - Biçimlerde bir evrimleşme değil, yer değiştirme olduğundan çizimler üst üste konduğunda, denemeler zor da olsa seçilebilir.

Şekil 3.3 Vaziyet planı üzerinde konut yerleşimi çalışması / Eskiz ve CAD programındaki çalışmaların karşılaştırılması



Şekil 3.4 Öneri Yazılımla Vaziyet planı üzerinde konut bloğu yerleşimi

Her konut bloğu çizilirken ekranın nesne yaratım alanına (Şekil 3.4'deki yeşil nokta) dokunarak bu çizilen biçim ayrı bir nesne olarak tanımlanacak ve çizim sonrasında bu nesneler parmak hareketleri ile ekranda sürüklenebilecek veya yönelimleri değiştirilebilecektir. Kullanıcı bu metodu kullanarak çeşitli yerleşim kompozisyonu çalışacak ve ekrandaki bir alana dokunarak yaptığı bu kompozisyon değişiklikleri kaydedip daha sonra tekrar oynatabilecektir.

Ekranda kayıt alanına (Şekil 3.4'deki kırmızı nokta) dokunduğunda , dokunduğu süre içinde bloklarının yönelimleri ve yerleri hakkındaki birbirlerine göre izafi değişiklikler kaydedilecek, kayıt sonrasında blokların birisinde yapılan yönelim ve yerindeki değişiklik, diğer bloğu da , bir önceki kaydedilen harekete bağlı kalacak şekilde, etkileyecektir (Şekil 3.4).

3.1.2 İletişim veya Kayıt Amaçlı Eskiz Çalışması

Kağıt kalem ile oluşturulan ifadelerin bir amacı da, ifadenin başkası tarafından veya oluşturanın kendisi tarafından daha sonra , bu durumunun zihinde tekrar oluşmasını sağlamaktır. Bu zihinde tekrar oluşturma işi için gereken ürünü kullanıcı kağıda not almanın yanında bilgisayarda bir yazılım üzerine çizerek/yazarak veya bir ses/video kaydedici üzerine de yaptığı işi anlatan bir kayıt yaparak da gerçekleştirebilir.

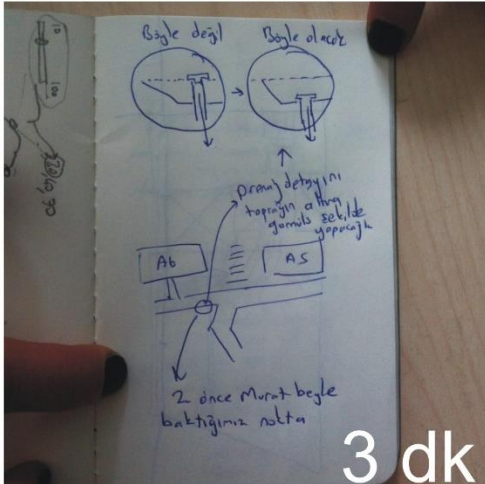
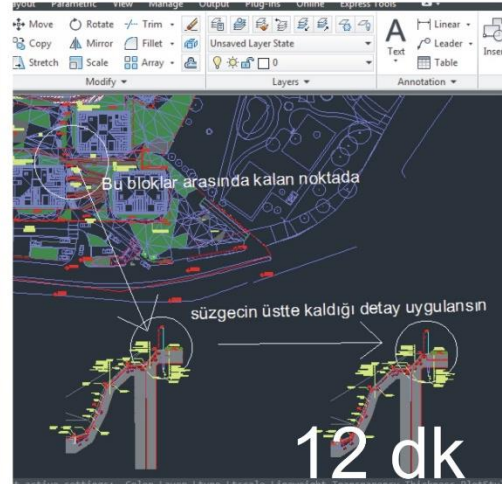
Aşağıda eskizin bahsi geçen kullanımına ilişkin uygulamaya örnek olarak iki farklı senaryoda, kağıt kalem ile ve bilgisayarda iki boyutlu drafting programı (Autocad) ve video kaydedici yazılımla ortaya çıkan iki farklı denemenin sonuç ürünleri sunulup, aşamalandırılıp, süreç ve sonuç ürün bazı temel nitelikleri bakımından karşılaştırılmaktadır. Daha sonrasında ise bu iki farklı yöntemin olumlu görülen niteliklerini kapsayabilecek alternatif üçüncü bir yöntem önerilecektir. Bir tablet bilgisayar üzerinde çalışacağı var sayılan bu üçüncü yöntem tez çalışmasının amacı olan, eskiz yöntemi ile bilgisayarın kabiliyetlerinin birbirini örtmeden beraber kullanımı çerçevesinde düşünülmüştür.

3.1.2.1 Çizim üzerinden iletişim kurma senaryosu

Bu senaryodaki birinci kullanıcı bir ikinci kullanıcıya fiziksel eskiz yoluyla ve Autocad yazılımıyla teknik bir bilgi tarif etmektedir. Birinci kullanıcı olarak bir inşaat sahasındaki teknik adam, ikinci kullanıcı olan imalat sorumlusu diğer teknik adama uzaktan, bir imalat detayındaki değişikliği (bir drenaj detayında değişiklik), bir görsel gönderilme yoluyla anlatacaktır.

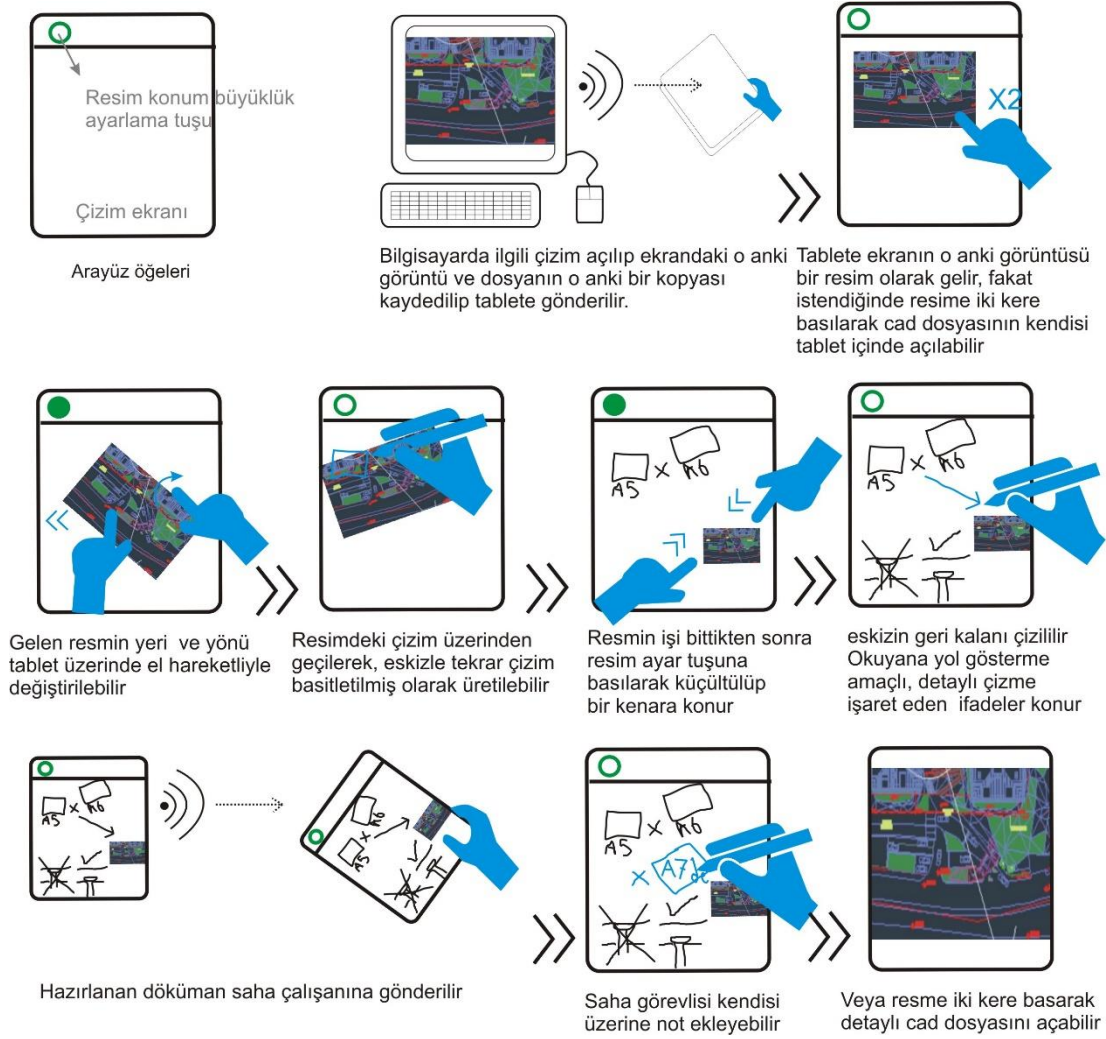
Birinci denemede mesajı anlatan görsel, gönderen tarafından ilgili iş alanında çok kullanılan çizim programı (Autocad) ile oluşturulmakta, ikinci denemede ise kalem ile kağıda çizilerek oluşturulmaktadır. Senaryodaki iki kişinin de ilgili imalat ile daha önceden beri beraber çalışan ve birbirlerinin diline ve sahadaki duruma hakim kişiler olduğu varsayılmaktadır. Bu iki deneme sonucu çıkan ürünler fotoğraflanmış, harcanan zaman ölçülmüş ve iki yöntemin ve sonuç ürünün farklı niteliklerinin karşılaştırılması tablo lanmıştır. **Şekil 3.5**'de karşılaştırılan iki yöntemin birbirinden daha kolay olmasını sağlayan ve sonuç ürünün farklı okumalara izin vermesi hususunda öne çıkan nitelikleri:

- Bilgisayarın öne çıkan niteliği olarak; aktarılan bilginin tarif şekli çok detaylı olduğu için, okuyan kişinin o konuyla çok ilgili olmasa bile anlama ihtimalinin daha fazla olması. Aktarılan bilginin bilgisayarda üzerinde tekrar çalışılabilecek bir drafting (Autocad) dosyası olması.

Sonuç ürün	Kağıt kalem ile 	Bilgisayar / Autocad ile 
Yapım aşamaları	○ Kağıt ve kalem edinmek ○ Arasındaki ilişkiyi anlatacağımız iki detayı basitçe anlatan çizim ve metin ○ İki detayın ilişkisini anlatan çizim ve metin ○ Yapılan eskizi cep telefonu ile fotoğrafını çekip göndermek	○ İlgili detay çiziminin açmak ○ Çizimin ilgili kısmını bulmak ve çalışılabilecek ölçekte yaklaşmak ○ Çizgi aracıyla ok işaretini çizmek ○ Metin yazıp yön ve yazı boyutu belirlemek ○ Çizimin kopyalanıp, vaziyet planı çiziminin yanına yapıştırma ○ Metinle ilgili çizim parçalarını daire içine almak için daire aracını açmak ve merkez noktasını ve çapını belirtmek ○ Çizilecek daireye fark edilebilecek şekilde renk ve çizgi kalınlığı verilmesi ○ Dökümanı kaydetmek ve e-posta ile göndermek
Nitelik	○ İlk bakıldığında anlaşılır ○ Eğer çizimde bahsi geçen detay, okunan tarafından iyi bilinmiyorsa anlaşılabilir ○ Çizim üzerinden ölçü alınmaz	○ Anlaşılması uzun sürer için, çizimin bazı kısımlarına yaklaşmak ve diğer alakasız detaylardan ayıklanması gerekir ○ Çizimde bahsi geçen detay, okunan tarafından iyi bilinmesi bile, herkes tarafından anlaşılabilir ○ Çizim üzerinden ölçü alınabilir, ölçekli çıktısı alınabilir

Şekil 3.5 Şantiyede haberleşme için iki farklı methodun, zaman, yapılış şekli ve nitelikleri açısından karşılaştırılması

- Eskizin öne çıkan niteliği olarak; aktarılan bilginin tarifi çok daha hızlı ve okunaklı yapılabilmesi. Bilgiyi alan kişinin gönderen kişiyle konuyla ilgili aynı ön bilgisi olma durumunda tarifi çok net yapılmamış olmasına rağmen yeteri kadar anlaşılabilmesi



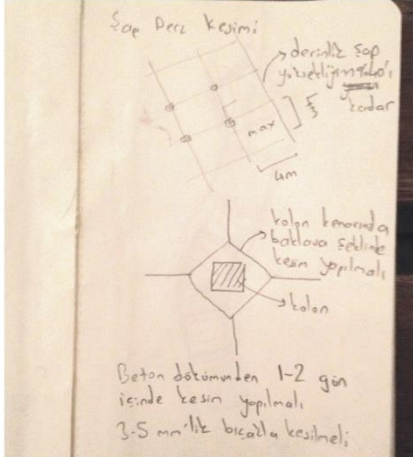
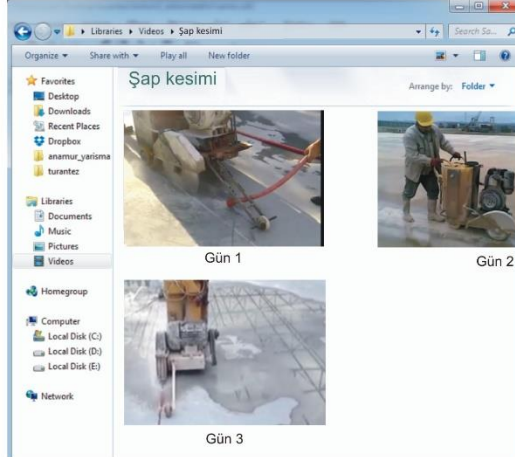
Şekil 3.6 Öneri Yazılımla şantiyede detay değişikliği hakkında tablet bilgisayarlarla haberleşme şekli

Yukarıda bahsi geçen niteliklerin beraber sağlanması amaçlayan üçüncü yolda saha sorumlusuna tarif yapan kişi bilgisayar üzerindeki hazır Autocad çizimini tabletine aktararak üzerinde serbest el çizim yapmakta ve çizerek hazırladığı bu dökümanı saha sorumlusunun tablet bilgisayarına göndermektedir. Burada tarifi yapan kullanıcı, önce ilgili detayın çizim programındaki (autocad) teknik çizimini açıp, ilgili noktaya ait

anlık ekran görüntüsünü ve dosyanın o anki halini tek bir tuşa basıp kaydedecek¹, daha sonrasında kağıt üzerine not alır gibi tablet üzerinde durumu anlatan eskizi çizip, ilgili detaya ait teknik çizim parçasını etiketleyecektir (**Şekil 3.6**). Bu sayede karşı taraf tabletini açtığında hem mesajı çabuk anlayacak, hem de gerekirse ölçekli-detaylı çizime ulaşabilecektir.

3.1.2.2 Çizim ile kayıt tutma senaryosu

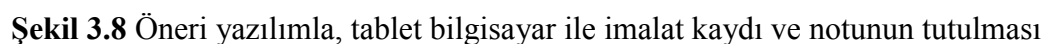
Bu senaryoda, inşaat sahasındaki bir imalatı (şap kesimi) izleyen kullanıcı, uygulanan imalat yöntemini daha sonra tekrar okumak/izlemek amacıyla kaydetmeyi amaçlamaktadır. Birinci denemede imalat, mobil telefon vasıtasıyla videoya alınarak (videoya alınırken de bir yandan görüntüyle açıklanamayan hususlara kayıt düşmek için, videoya alan kişi konuşmaktadır) ve fotoğraf çekilerek kaydedilmektedir.

Sonuç ürün	Kağıt kalem ile 	Video kaydedici ve bilgisayar ile 
Yapım aşamaları	- Şap kesiminin yapıldığı üç gün boyunca, yapım methodu, malzeme ve ölçüler ile ilgili bilgilerin basit çizimler üzerinde anlatılarak deftere kaydedilmesi. - Şap kesimi hakkında birşey hatırlanmak istendiğinde defter üzerinde ilgili notların bulunarak okunması. - Şap kesiminin yapıldığı üç gün boyunca imalat yerinde video çekilip, videoda görülemeyen malzeme ve ölçü gibi bilgiler video esnasında ağızla söylenerek kayıt edilmesi. - Bu sesli video kayıtların daha sonrasında bilgisayara içeriğine dair fikir verecek isimlerle kaydedilmesi. - Şap kesimi hakkında birşey hatırlanmak istendiğinde bu kayıtların baştan sona oynatılarak izlenmesi.	
Nitelik	- Bir yandan imalat izlenirken, bir yandan not tutulduğu için bazı noktalar kaçırılabilir. - Daha sonrasında, tutulan notlar çok çabuk şekilde anlaşılır - Tutulan notlarda bir bilginin tekrarı, veya ayıklanması gereken gereksiz bir bilgi bulunmaz - İmalat görüntülediği için, video çekiminde akla gelmeyen ayrıntılar bile daha sonrasında izlenebilir. - Daha sonrasında, çekilen videoları baştan sona kadar izlemek uzun zaman alır. - Videolarda birçok tekrar ve ayıklanması gereken gereksiz bilgi bulunabilir	

Şekil 3.7 Bir imalat şeklinin kaydedilmesi / Kağıt ve dijital ortamda yapılan kayıtların karşılaştırılması

1 Bu senaryoda çizim programında, o anki ekran görüntüsü kaydedilip ve çalışılan dosyanın bir kopyasının alınabileceği bir yazılım parçası olacağı varsayılmıştır.

- Bilgisayarın öne çıkan niteliği olarak; video kaydı yapıldığı için imalatın her anının ve detayının kaydedilmesi, bu sayede daha sonra bu kayda bakıldığında her detaya ulaşılabilmesi.



- Eskizin öne çıkan niteliği olarak; imalatın kaydedilen kısımlarının daha sonra çabuk şekilde ayırt edilebilip ve okunabilmesi (video kaydında ise imalat kısımlarının hepsinin sırayla uzun sürede izlenmesi gerekmektedir).

Yukarıda bahsi geçen niteliklerin beraber sağlanması amaçlayan üçüncü yolda imalatın video kaydı bir tablet bilgisayarla alınıp, bu video kayıtları üzerine not alınabilmektedir. Üçüncü yol olarak önerilen yazılımda kullanıcı, tableti ile kağıt üzerinde yaptığına benzer notlar alabildiği gibi imalat sırasında video ve fotoğraf çekebilmekte, bu görselleri tablet ekranı üzerinde istediği yere koyup boyutlandırabilmektedir (**Şekil 3.8**). Daha sonrasında kullanıcı, videoyu istediği anında dondurarak, bu görüntü üzerinden geçerek kendisi kalemle çizim yapabilmektedir. Böylece imalat anında kalemle not alırken kesintisiz izleyemediği imalat anını kaçırmamakta, ve daha sonrasında da eskiz ile ürettiği görseli yetersiz görmesi durumunda videoyu tekrar oynatabilmekte veya fotoğrafı inceleyebilmektedir. Eskiz ile yaptığı kısım sayesinde, video daha sonra ne olduğu kolayca anlaşılır hale gelecektir.

3.1.3 Bilgisayarda üretilen eskizin, hipermetin kavramıyla kurgulanması / Memex, internet ve eskiz

Bilgisayar ortamında oluşturulan derlenen her bilgi/içerik, çeşitli şekillerde bilgisayar ortamında istiflenir. Bu istifleme şekli bu bilginin/içeriğin nasıl okunacağını, ulaşılabileceğini etkilediği gibi, derlenen bilginin de bu istiflemesine uygun şekilde oluşturulması gerekir. Bu duruma tersten bakıldığında derlenen birçok bilginin, kurgularındaki benzerlikler ve uyduğu bazı standartlar, bu istifleme şekillerinin ne olacağını da etkiler. Bu oluşturulan içerik ve istifleme şekli ilişkisi önemli olduğu için, önerilen eskiz yazılımına, bilgisayardaki veri istifindeki önemli kavramlardan biri olan hipermetin kavramı ve öncülü Memex penceresinden de bakılacaktır.

1945 yılında Vannevar Bush, as we may think isimli makalesinde Memex (Memory-index) isimli bir araç-kavram dan söz eder. Ağaç diyagramına benzer şekillerde iç içe gruplandırılmış bilgi yığınları arasından bilgiyi bulmak yerine, her bilginin okuyucunun kafasında oluşturduğu çağrışımlar sonucu ilişki kazanan bilgi rotası üzerinden bulunmasını sağlayan için bir araç önerisidir (Bush, 1945).

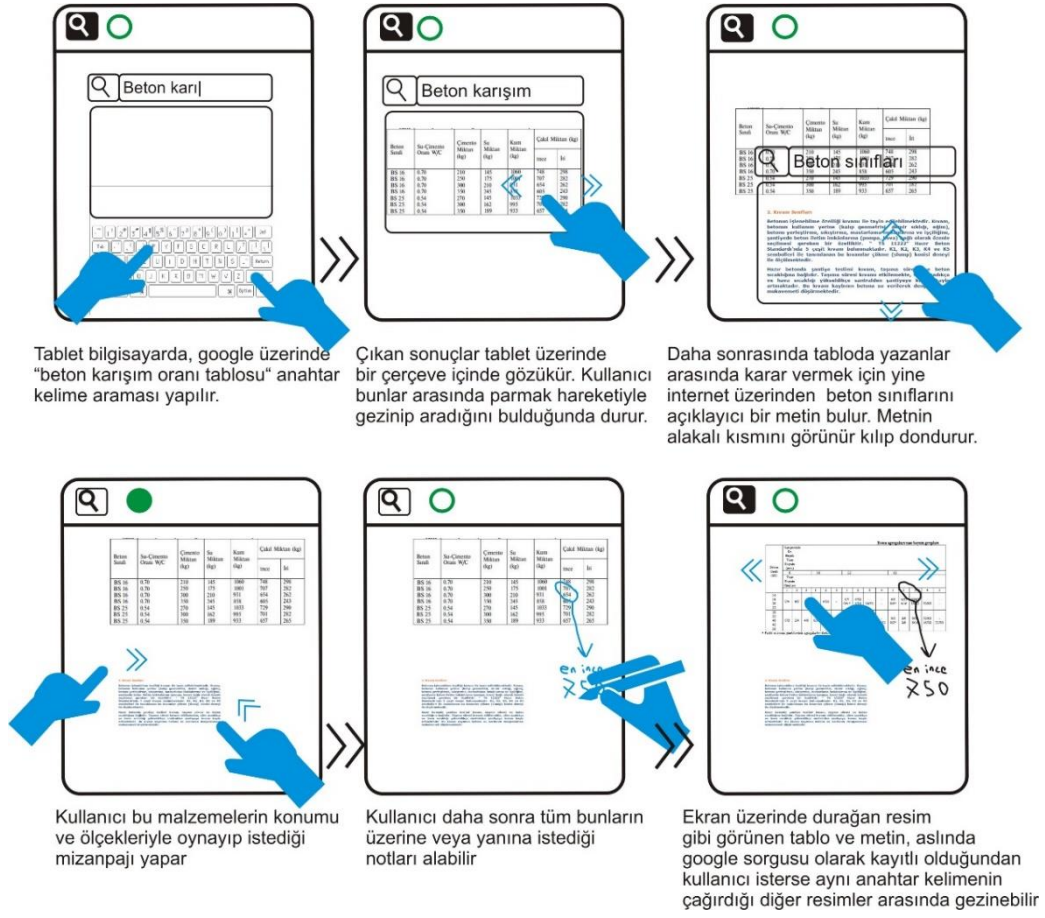
Önerdiği alet, binlerce kitabın sayfa fotoğraflarının mikro filmlerle saklanıp, kullanıcı tarafından okunabilmesi, daha da önemlisi bu sayfaların kopyalanıp, kullanıcı tarafından eklemeler yapıp, tüm bunların kullanıcının araştırdığı bilgi için oluşturduğu rotada yer alabilmesini sağlar. Kullanıcının bir anahtar kelime/kavram hakkında birçok kaynak ve kendi eklediği eskizlerle oluşturduğu rota başka birisi tarafından da okunabilir, rota boyunca karşısına çıkan malzemenin asıl kaynak sayfalarına da ulaşılabilir.

Memex ile kullanıcı çeşitli kaynaklardan elde ettiği metin ve fotoğrafların yanısıra kendi notları ve çizimlerini de, oluşturduğu bilgi rotasında kullanır. Başka bir açıdan bakılırsa aslında oluşturduğu rota, başka ortamlardan bilgilerin de bulunduğu, ilgili anahtar kelime/kavram'ı anlatan bir bilgi kolajı olarak da görülebilir. Ve bu bilgi kolajının mizanpajının kullanıcının yapabilmesi ve kalemle kendi çizim ve notlarını alabilmesi, bazı kullanım senaryolarında bu kolaja eskiz mahiyeti kazandırabilir.

Aşağıdaki kullanım senaryolarında anlatılan yazılım önerisi de birçok yönden Memex'e benzetilebilir. Kullanıcı yazılımı kullandığı donanım üzerinden girdiği ve internet üzerinden çağırdığı bilgileri, tekrar internet üzerinde kaydederek, bu oluşturduğu eskizin basitliği ve anlaşılabilirliğine sahip ilişkisel kolajın, başkaları tarafından da çağrılıp okunup, kopyalanıp bundan başka bir ilişkisel kolajın yaratımını sağlayabilecektir. Şayet bu anlatılan kolaj-eskizle bilgi saklama pratiğinin çok kullanılması halinde Google gibi görsel ve metin bazlı arama motorlarında da, kolaj-eskiz biçimindeki bilginin aranması ve içinde dolaşımı sağlanabilir .

3.1.3.1 Mevcut bilgi üzerine eskizle eklemeler

Kullanıcı bu örnekte mevcut bir bilgi üzerine eklemelerle kendi işine yarayan eskiz-kolaj üretimi yapmaktadır. Beton bahçe duvarı için malzeme siparişi vermek isteyen kullanıcı, bu senaryoda google üzerinde “beton karışım oranları tablosu” isimli resim araması ve “dış mekanda hangi tip beton kullanılmalı” isimli metin araması yapmakta ve sonuç olarak çıkan, işine yaradığını düşündüğü tablo ve internet sitesi metnini tablet bilgisayarındaki yazılıma yüklemektedir. Metni okuyup tabloda uygun gördüğü beton tipini seçtikten sonra, bu tabloyu tablet bilgisayardaki yazılımda altlık yaparak, miktar hesabı ve malzeme siparişi esnasında aldığı notları bu altlık üzerine kalemle not olarak yapmaktadır (Şekil3.9).



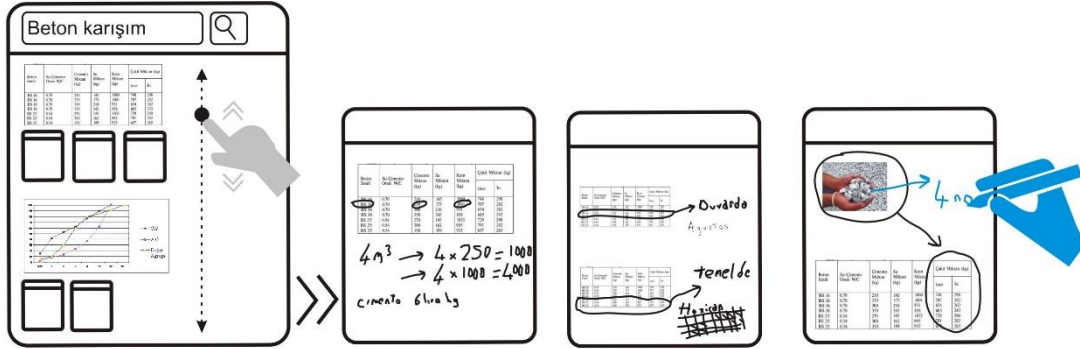
Şekil 3.9 Öneri yazılımla, internet üzerindeki bilgiyi kullanarak kolaj-eskiz oluşturma

Daha sonrasında da bu notu saklamakta, isterse daha sonrasında internet araması şeklinde yazılımda kayıtlı duran tablo ve metnin aynı anahtar kelimede çıkan başka sonuçlarına da eskiz sayfası içinden ulaşabilmektedir. Kullanıcı tüm bunların yerine bilgisayar ekranında tablo açıkken, kâğıt ve kalem ile not tutabilirdi, fakat bu tuttuğu not daha sonrasında yukarıda anlatıldığı gibi kullanılamazdı.

3.1.3.2 Aynı Mevcut Bilgi Üzerine Yapılan Eskizlerin Beraber Okunması

Bu senaryoda, farklı kişilerce aynı mevcut bilgi üzerine oluşturulan eskiz-kolajların yanyana okunduğu bir durum anlatılmaktadır. Kullanıcı olan, yapının neresinde hangi beton karışım oranları kullanılması hakkında bilgi sahibi olmak isteyen bir öğrenci, google üzerinde eskiz-kolaja malzeme olmuş resimler arasında "beton karışım oranları tablosu" anahtar kelimesini aratmış, ve resimler arasında en çok eskiz-kolaja malzeme olmuş resmin bulunduğu eskiz-kolajları incelemektedir.

Karşısına çıkan sonuçları sırayla incelemekte, araştırdığı konu hakkında bulduğu resmin kopyalarının, insanlar tarafından nasıl çok farklı işlerde kullanıldığını görüp, fikir sahibi olmaktadır (Şekil 3.10).



Tablet bilgisayarda, google üzerinde "beton karışım" anahtar kelimesiyle resim araması yapıldığında çıkan sonuç resim listesi altında, bu resmi kullanan eskiz-kolajlar da gözükür

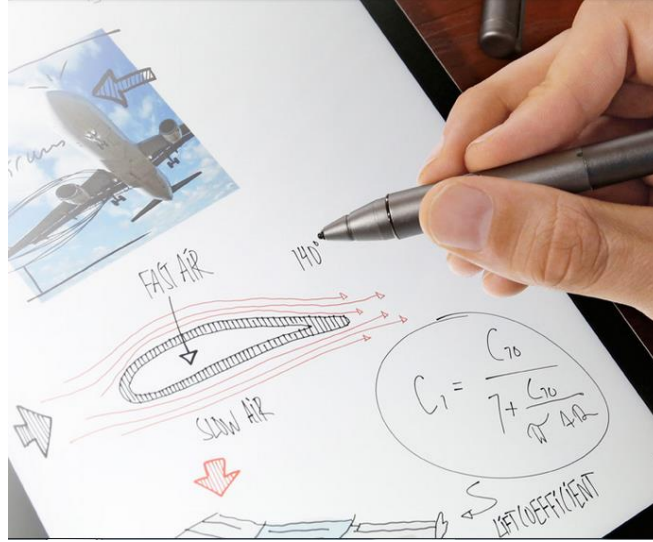
Kullanıcı sorgu sonucu çıkan ilgili resmin, kaynak olarak kullanıldığı eskiz kolajları inceleyebilir. Başka kullanıcıların birçok farklı şekilde, bu resmi nasıl kullandıklarını görür, bunlara kendisi de ekleme yapabilir, değiştirebilir

Şekil 3.10 Öneri yazılımla, internet üzerindeki aynı malzemeyi kullanarak oluşturulmuş kolaj-eskizlerin beraber okunması

3.2 Yazılım Modeli

Bu bölümde anlatılacak yazılım modeli, kalem ve kağıdın kullanım pratiğini engellemeden bilgisayarların bu kullanım pratiğine ne gibi yeni nitelikler kazandırabileceği hakkında, daha önce ortaya atılmış fikirler ile tez metni yazarının kendi tecrübelerinden yararlanarak oluşturduğu bir öneridir. İleriki sayfalarda detaylıca anlatılacak yazılım önerisini kısaca açıklamak gerekirse, tablet bilgisayar ekranı üzerinden dijital kalem ve parmak girdisi ile kullanılan, her zaman alışlagelmiş boş kağıdın üzerine kalemle eskizin yapılabildiği fakat istenildiği zaman çeşitli bilgisayara has araçların kullanabileceği bir kurgudur.

Bahsi geçen yazılım modelinin üstünde çalışacağı varsayılan donanım, basınç hassasiyetini kendi hissedip bilgisayara bluetooth üzerinden gönderebilen bir dijital kalem ve dokunmayı algılayan ekrana sahip, video-fotoğraf çekebilene ve internete bağlanabilene herhangi bir tablet bilgisayardır (Şekil 3.10). Bu yazılımın daha iyi çalışabileceği düşünülen, şu an mevcut fakat daha tablet bilgisayar üzerinde yer edinememiş teknolojiler 4.2.2. numaralı bölümde anlatılmaktadır.



Şekil 3.10 Baskı hassasiyetli dijital kalem ve tablet bilgisayar (Url4)

Öneri modelin yazılım kodu kısmi şekilde yazılabilmektedir. Bu yüzden bu bölümdeki tasvirler kurgudur, tez sonunda ekte yazılımın bazı fonksiyonlarının çalıştığı Java-Processing’de yazılmış kısmi kodlar bulunmaktadır. Model anlatılırken ilk önce model ilkeleri ve bu ilkelerin yazılım arayüzüne ve nesnelere nasıl yansıdığı anlatılıp daha sonrasında programın kullanımının canlandırılması şeklinde sıralı resimlerle anlatım yapılacaktır. Daha sonrasındaki kısımda ise yazılımın kod kurgusu/altyapısı üzerinden daha teknik bir anlatım olacaktır.

3.2.1 Modelin ilkeleri

Yazılım modelinin ilkeleri sıralacak olursa;

- 1- Defter ve kalemle ne yapılıyorsa, bu eylemlerin kesinlikle perdelenmeden, her an yazılım üzerinde yapılıyor olabilmesi (serbest el çizim, elin yazarken ekrana dayanabilmesi, kalemin bastırıldığında daha koyu renk vermesi, silme, kağıt üzerine cetvel koyarak çizme)
- 2- Ekrana çizilen grafik nesneler dışında yazılım arayüzün çağrılmadığı zaman kesinlikle gözükmemesi, ekranda bir bölümlendirme olmaması.
- 3- Bu iki ilkeyi örtmeyecek şekilde, eskiz ile ilişkisel bilgi ifadesi ve üretimine yardımcı olacak, kolay kontrol edilebilir temel yazılım araçlarının kullanılabilir olması

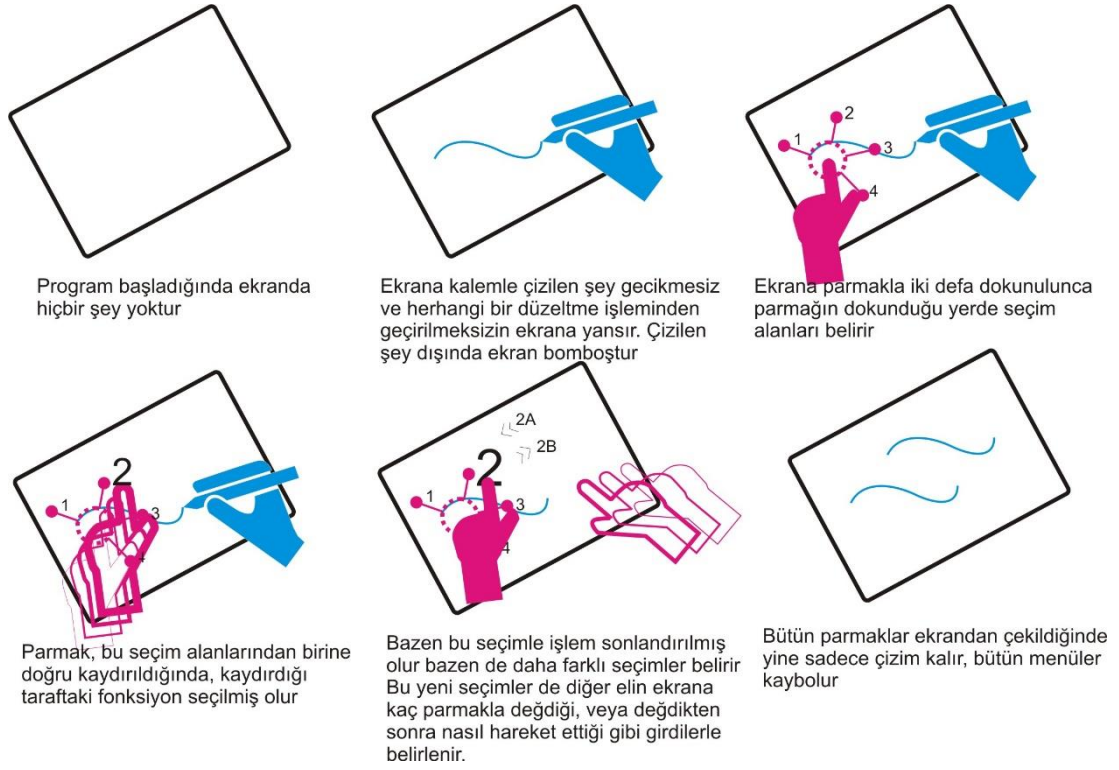
Burada üçüncü maddede bahsi geçen eskiz ile ilişkisel bilginin ifadesi ve üretimine yardımcı olacak araçlar-eylemlerden bu modelde gerçekleştirilmeye çalışmaları;

- + Eskizin geçmişin kaydedilip, kullanıcı tarafından daha sonra kontrollü olarak çağırılabilmesi
- + Ekranda belli bir referans noktasına bağlı, kopyalanabilen, hareket ettirilebilen grafik nesnelerin oluşturulabilmesi
- + Bu grafik nesnelerin, pozisyon, renk gibi nitelikleri arasında ilişkiler tanımlanabilmesi
- + Yazılım içinde, ekranda kullanıcı tarafından açılan bir delikten daha başka bir yazılımın ekran çıktısının görüntülene bilmesi, ve bu delikten kontrol edilebilmesi
- + Eskiz bilgisinin internet üzerinde kayıtlı olması, ekranda açılan delikten internet üzerinden bir bilgi çağırabilme, ve bu çağırılan bilgiyle yapılan eskizin birbirine bağlı olarak internet üzerinde, başkası tarafından da ulaşılabilinecek şekilde saklanabilmesi

İkinci maddede bahsi geçen arayüz, çizimi perdelememek için ancak parmakla ekrana iki defa ard ardına dokunulduğunda, dokunulduğu yeri merkez kabul ederek çıkan ve daha sonrasında bu dokunan parmağın sürüklenmesiyle kontrol edilip el çekildiğinde tekrar kaybolan saydam seçim alanlarıdır (**Şekil 3.11**). Kullanıcı ekranın istediği yerinde çağırabildiği için menüler, kullanıcının ekranın çizim olmayan bir tarafında menüyü çıkartması durumunda, hali hazırda yaptığı çizimin görünürlüğünü engellemez. Kalem ve parmak neredeyse tamamen birbirinden bağımsız çalışan girdi şekilleridir, bu sayede kalemin serbest çizimle olan ilişkisi hiç kesilmez. Parmak girdilerinin başlangıçta bir defa değil de iki defa ekrana değerek açılan menüden idaresinin nedeni de kalemle yazarken elin ister istemez ekrana dayanması sonucunda ortaya çıkan hatalı parmak girdisinin önlenmesidir.

Tekrar ikinci maddede bahsi geçen ekran bölünmesinin olmaması için, ekranda herhangi bir seçim alanı ve menü için yer ayrılmamıştır, bu sayede menülerin ister istemez içerikle oluşturduğu grafik ilişki engellenmiş olup, boş kağıda çizim metaforu güçlenmiştir.

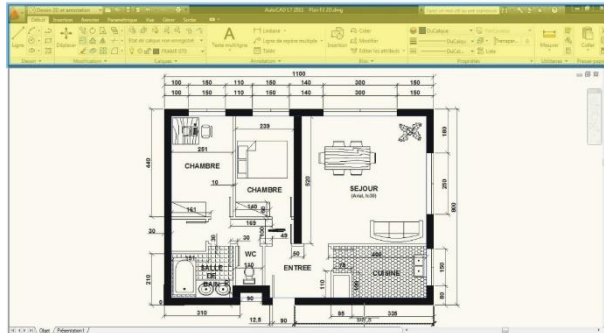
Yine boş kağıt üzerinde çizim algısının kaybolmaması için ekrandaki grafik nesneler veya başka programdan çağırılan grafiklerin, çerçeve, ayar çubuğu vb. öğeleri ancak, parmakla dokunma devreye girdiğinde görünür hale gelir. Parmakla ekrana dokunularak açılan seçim alanları olabildiğince az basamaklı olacak şekilde düşünülmüştür.



Şekil 3.11 Öneri modelin arayüzü ve seçim alanları hakkında sıralı ekran çıktıları örnekleri

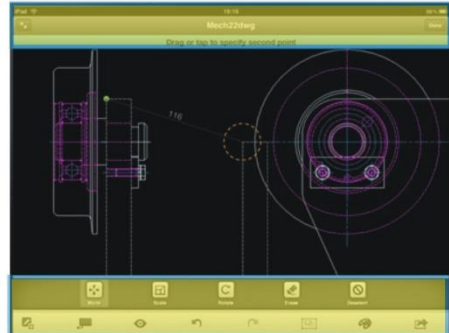
Bunun nedeni açılarak ekranda büyük yer kaplayan menülerin çizimin görünürlüğünü engellemesi ve hatta açılan menülerin çok uzun olması durumunda ekrana sığmayabilmesidir. Bunun için kullanıcı tarafından ezberlenmesi gereken belli bir mantıkta gruplanmış parmak hareket kombinasyonları ile seçimler yapılır. Bu ilk kullanıcı için zorluk oluşturabilir fakat ileriki zamanlarda daha hızlı kullanabileceği ve daha geçirgen bir arayüze sahip olur.

Autocad PC (masaüstü bilgisayar)



Ekranın %18'inde 80 seçim alanı

Autocad iOS (tablet bilgisayar)



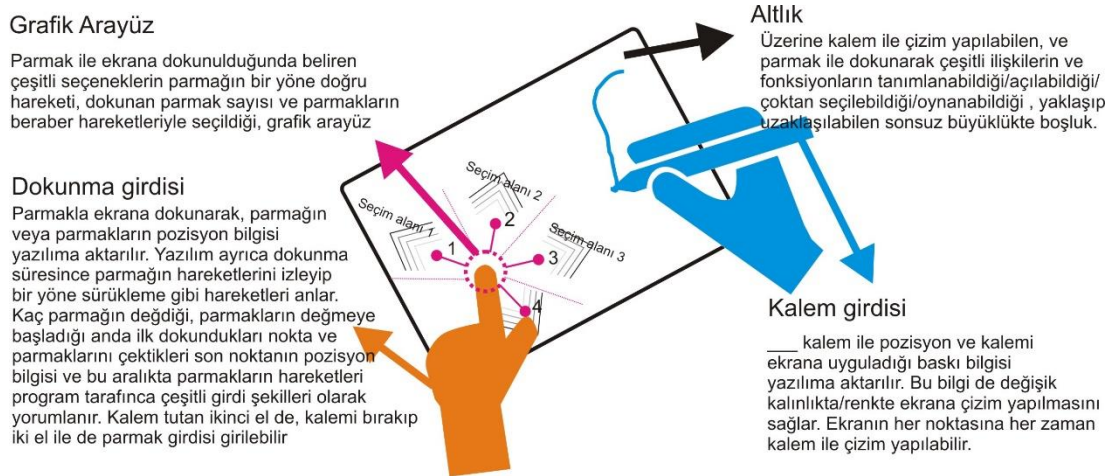
Ekranın %30'unda 18 seçim alanı

Şekil 3.12 Fare ile kullanılan masaüstü bir bilgisayar ekranıyla, parmak ile pozisyon girilen tablet bilgisayardaki yazılım arayüzlerinin kapladığı alan karşılaştırması

Bu geçirgenlik tablet bilgisayar gibi zaten ekranı küçük ve fare gibi hassas bir işaretçi yerine parmak ile çok daha az hassas bir şekilde girdi aldığından seçim alanları büyük olmak zorunda olan bilgisayarlar bilgisayarlar için çok önemlidir (Şekil 3.12).

3.2.2 Modelin kullanım şekli üzerinden anlatılması

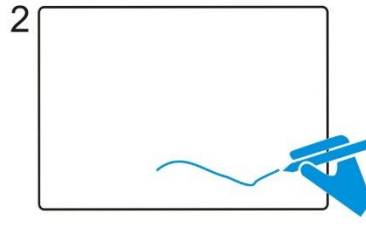
Bu kısımda yazılım modeli kullanılırken kullanıcının gözünden sıralı ekran görüntüleri ve o görüntüleri açıklayan metinler ile anlatılacaktır (Şekil 3.14). Şekil 3.13’de sıralı ekran görüntülerinde görülecek temel öğeler açıklanmıştır. Kullanıcı iki elindeki parmaklar ve bir elinde tuttuğu dijital kalem ile programı kullanmakta, genellemek gerekirse dijital kalem ile ekrana bir şey çizip , parmaklarıyla dokunarak veya sürükleyerek çeşitli seçimler yapmakta ve bilgisayar fonksiyonlarını tetiklemektedir. Ekran görüntülerini anlatan şemalarda mavi renk yeni girilen kalem girdisini, pembe renk yeni girilen parmak girdisi ve ekrana getirdiklerini, siyah renk ise zaten var olan grafik nesneleri göstermektedir



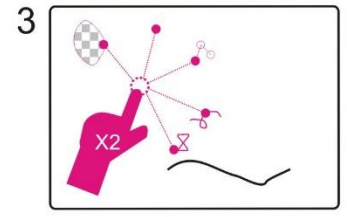
Şekil 3.13 Modelin kullanıcıyla ilişki kurduğu temel öğeler



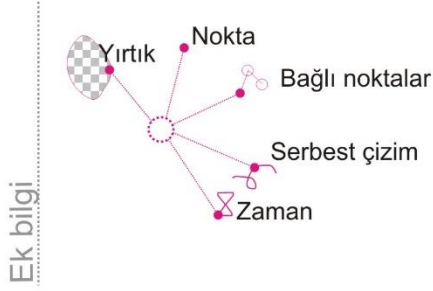
1 Yazılım açıldığında ekranda hiçbir görüntü, klavuz çizgi veya arayüz grafiği yoktur.



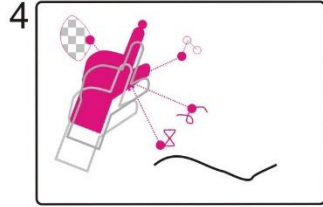
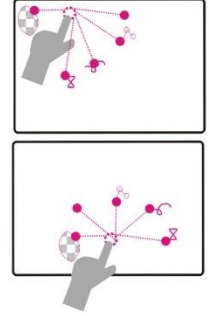
2 Kullanıcı, dijital kalemle ekrana, hiçbir şeyi seçmeden hemen çizmeye başlayabilir



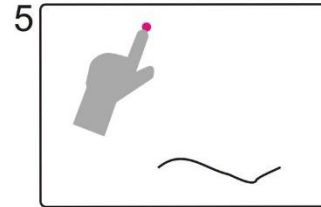
3 Ekrana parmağı ile iki defa ard ardına dokunduğunda, parmağın dokunduğu nokta etrafında yeni nesne menüsü açılır.



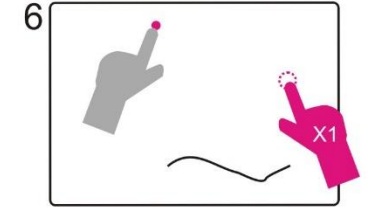
Bu menüde kullanıcının karşısına beş nesne tipi çıkar (bir referans noktasına bağlı yırtık, nokta, bağlı nokta, serbest çizim ve zaman). Menüü açmak için ekrana iki kere değilen noktadan çıkan grafik arayüzdeki beş dal ekranın görünür yerlerine doğru olur, yani sağdaki örneklerdeki gibi ekranın değişik yerlerinde, menünün görünürlüğünün sağlanması için farklı dallanmalar olur



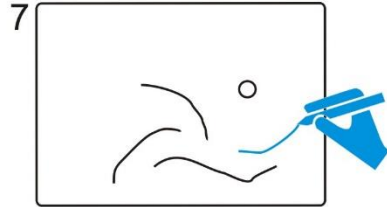
4 Parmak çekilmeden bu menüde çıkan beş koldan hangisine sürüklenirse, o nesne tipi seçilmiş olur



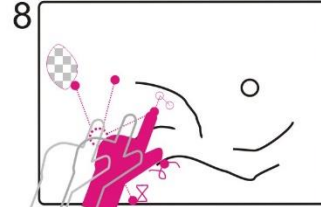
5 Parmak bu örnekte nokta nesnesini seçmiştir, nesne seçildikten sonra menü kaybolur, parmak çekilmeden noktanın nereye konacağı girilmelidir



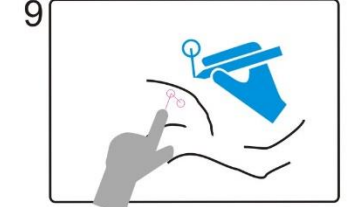
6 Diğer el ile nokta konması istenen yerlere birer defa değilerek noktalar konur



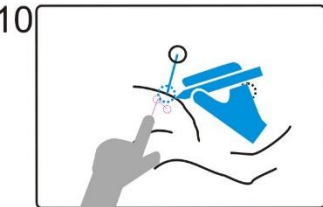
7 Ekrana bir tane nokta konmuştur. Daha sonrasında Kalemle serbest el çizim şeklinde eklemeler yapılır



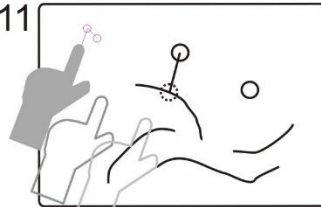
8 Daha sonrasında ekrana tekrar iki kez değilip, nesne menüsü açılıp bu sefer bağlı noktalar nesnesi seçilir



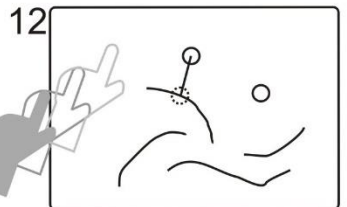
9 Bu nesne seçiliyken ekrana parmak veya kalemle (bu sefer kalemle giriliyor) değilip bağlı noktalar başlangıcı seçilir daha sonra kalemin kalktığı yer de..



10 .. Bu bağlı noktaların diğerinin yeri olur ve bu iki nokta arasında bir çizgi oluşur, bu şekilde uç uca çizgi çizilmeye devam edilebilir

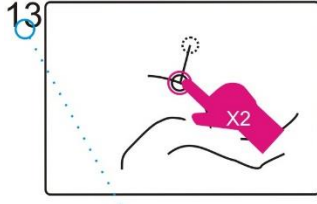


11 Menüü açtığından beri ekrana değmek zorunda olan sol el, ekranda hareket ettirildiğinde birşey olmaz, hareket ettirilebilir veya ...

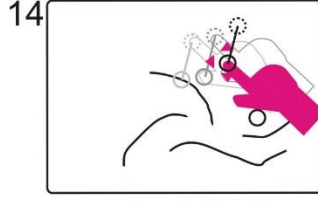


12 .. El yorulmuşsa veya birşeyi engelliyekse dışarıya doğru sürüklendiğinde seçim, ekrana iki kere art arda değinceye kadar seçili kalır.

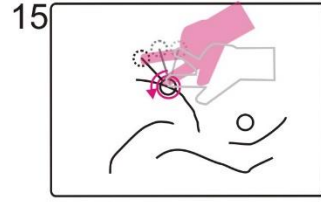
Şekil 3.14 Model yazılımın kullanım şeklini anlatır tablo



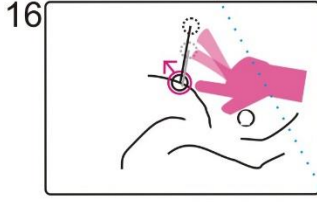
Bu bağlı noktalara iki defa üst üste değildiğinde, bu bağlı nokta nesnesinin kontrol noktası seçilmiş olur. Daha önceki gibi boşlukta iki defa ekrana değildiğinde çıkan nesne yaratım menüsü çıkmaz



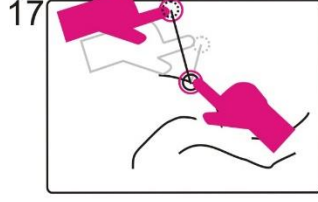
Tek parmak ile sürüklendiğinde nesne hareket eder...



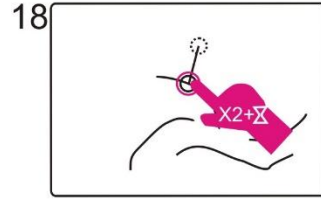
Bir parmak sabit, ikinci parmak aradaki uzaklık sabit kalacak şekilde dönüyorsa nesne döner ...



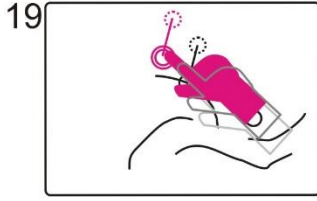
Bir nokta sabit kalacak şekilde ikinci parmak yaklaşıp uzaklaştığında, nesne büyür veya küçülür..



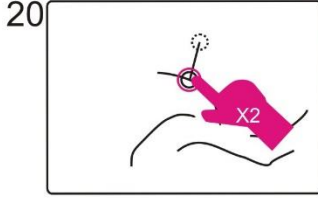
Bu hareketlerde ikinci parmak boşluğa değil de, bağlı diğer noktayı değerse, sadece diğer nokta hareket eder..



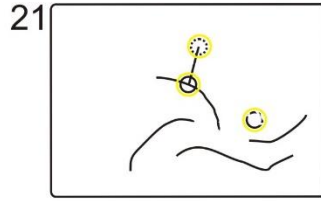
Bu bağlı noktalar nesnesi kopyalanmak istenirse iki defa üzerine değilip, basılı olarak bir süre beklenir ...



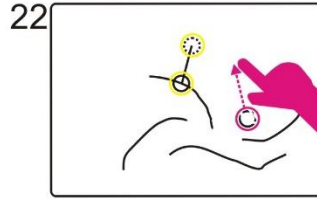
Parmek çekilmeden sürüklenip kopyalanmak istenen noktaya taşındığında parmak çekildiğinde nesne bu noktaya kopyalanmış olur



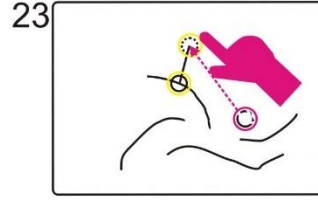
13. Basamaktan bu yanaki basamaklar bilgilendirme amaçlıydı, şimdi bu basamaktan tekrar devam edilmektedir



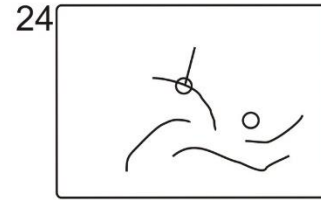
Kullanıcı iki defa bir noktaya değdikten sonra elini hareket ettirmeden çekerse ekrandaki tüm noktalar seçilebilir bir hal alır...



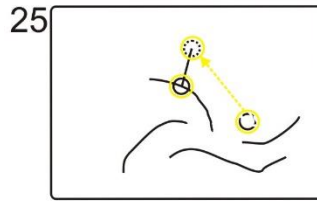
.. Kullanıcı bu noktalardan birine değip parmağını kaydırdığında bu noktadan bir ok çıkar, bu ok birbiriyle ilişkili nesneleri eşleştirmek için çıkmaktadır...



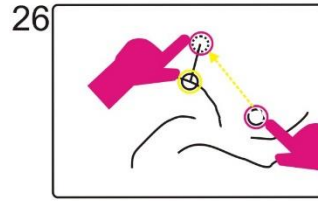
... Ok bağlı noktalar nesnesinden bir noktaya götürülüp parmak çekildiğinde bu noktalar bağlanır, daha sonra boşluğa parmakla değilip bu ilişkilendirme modundan çıkılır



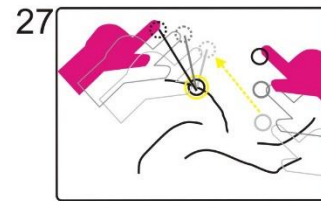
... İlişkilendirme bittikten sonra, ilişkilendirilen nesnelerden kaynak olanı görünür kalır, diğerlerinin noktaları silinir ve ulaşamaz...



13. Basamaktaki nokta üzerine çift değip parmağı hemen çekme işlemi tekrarlanırsa ilişkilendirme modu tekrar açılır. Tekrar başka bir ilişkilendirme yapılabilir veya mevcutlarda değişiklikler yapılabilir

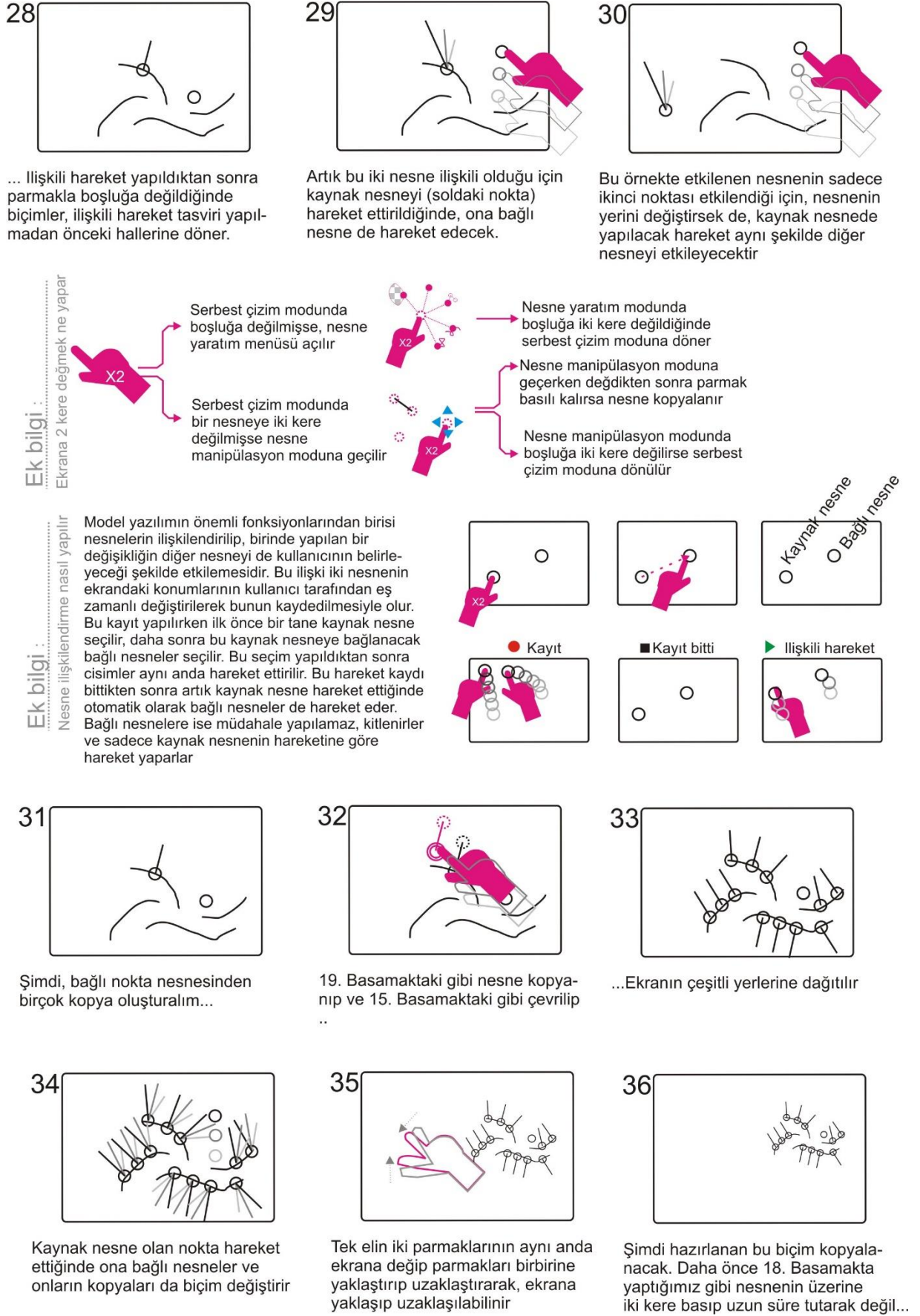


İlişkilendirme modundayken iki parmakla iki nesneye dokunulursa ve bu nesneler zaten birbirine bağlıysa ilişki tanımlama işlemi başlar...

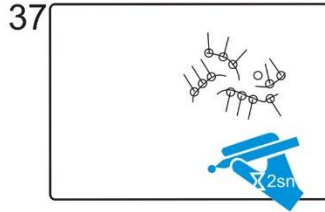


İki nesne beraber aynı anda, kullanıcının istediği gibi hareket ettirilir, yazılım bu hareketlere göre, hedef nesnenin, kaynak nesnenin hareketine göre nasıl hareket edeceğini hesaplar

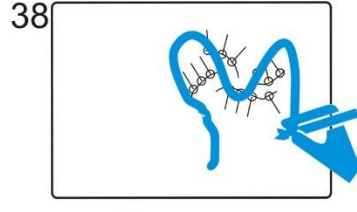
Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



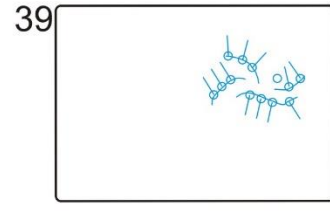
Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



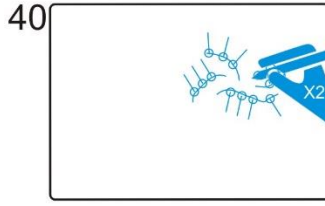
Şimdi ekrandaki tüm nesneler kopyalacak. Aşama 19 daki gibi tek nesne kopyalamayamayacağı için bu sefer kalemle çoklu seçim yolu uygulanacak. Ekran kalemle değip 2 sn beklenir..



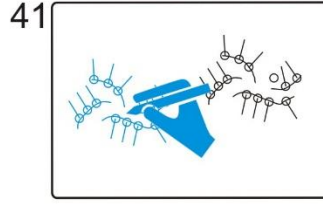
.. 2sn sonunda kalem izi kalınlaşır, kalem hareket ettirildiğinde artık kalemin geçtiği ize değen tüm nesneler seçilir



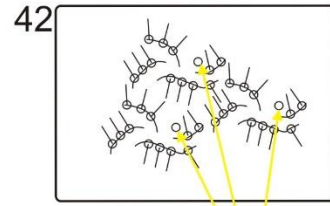
Kalem ekrandan çekildiğinde seçim tamamlanmıştır. Seçilmiş olan nesnelerin rengi değişir..



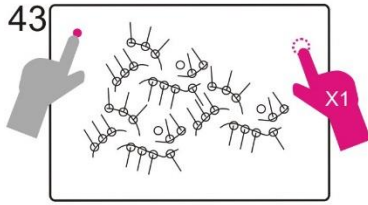
Kalemle seçili nesneler üzerine iki defa ardı ardına değerek seçili nesnelerin bir kopyası çıkarılır..



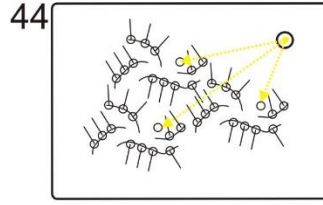
.. Kalemle sürüklenerek istenilen yere bırakılır



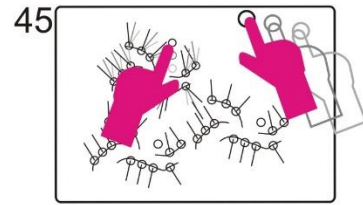
Bu yöntemle üç kopya oluşturulur. Şu an ekranda 3 grup bağlı çizgi ve bunları kontrol eden 3 tane nokta var



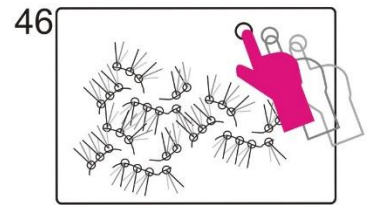
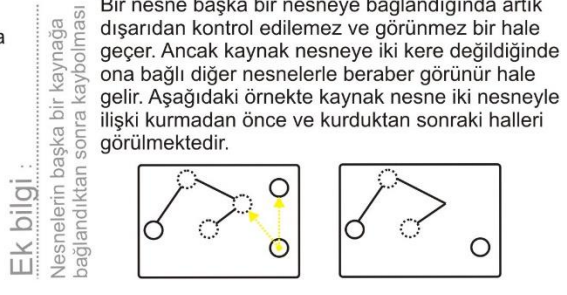
Şimdi bu üç kontrol noktasını başka bir noktaya bağlayıp aşama 34 daki gibi bir hareket bu sefer aynı anda üç grup için yapılacak. Ekranda bir nokta aşama 6daki gibi oluşturulur...



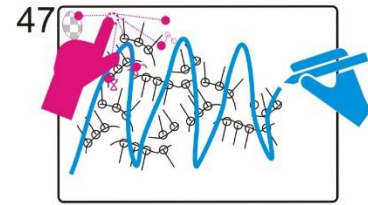
.. 20-25 aşamalar arasındaki işleme benzer işleme yeni konulan nokta diğer üç kontrol noktasına bağlanır



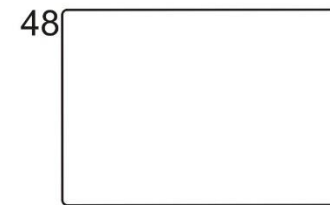
... 26 ve 27 aşamalarındaki işleme benzer şekilde yeni nokta ve üç nokta teker teker hareket ettirilerek ilişkilendirilir



İlişkilendirme tamamlanır. Aşama 29 daki gibi tek bir noktanın hareketiyle tüm doğru parçaları hareket eder

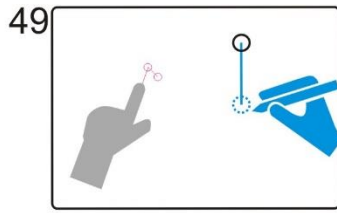


Şimdi tüm çizilenler silinecek, bunu yapmak için bir parmakla ekrana değilip kalemle silginin hareketine benzer şekilde silinecek nesnelerin..

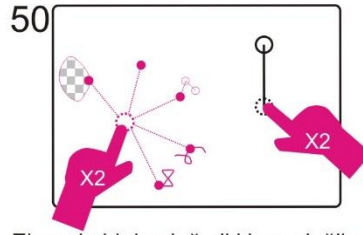


... Üzerinden geçilerek nesneler silinir

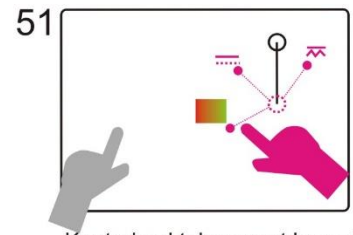
Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



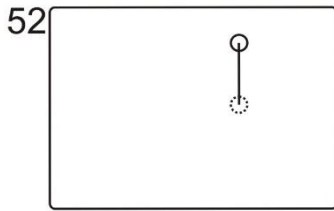
Şimdi bağlı noktalar nesnesinin nokta konumları dışında diğer değişkeni doğru parçası rengiyle dışardaki bir noktanın konumu ilişkilendirmesi yapı-lacak. İlk önce 8-10uncu aşamalarındaki gibi bağlı noktalar nesnesi tanımlanır..



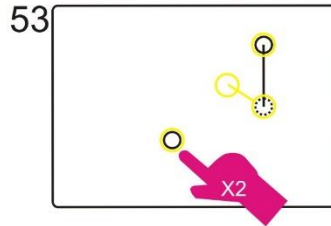
Ekranı bir boşluğa iki kere değil, yeni nesne menüsü açılır, bu menüde bir şey seçmek için parmağı sürükledikten sonra başa bir nesneyi iki kere değil, ikinci değeri nesnenin niteliklerini kontrol edeceğimiz yeni ...



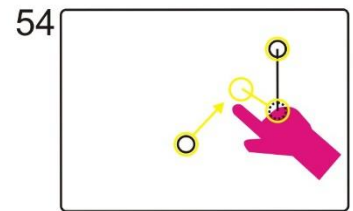
...Kontrol noktaları yaratılmış olur.
Bu örnekte bu nesnenin rengini
kontrol eden nokta, elin üzerine
sürüklenmesiyle seçiliyor



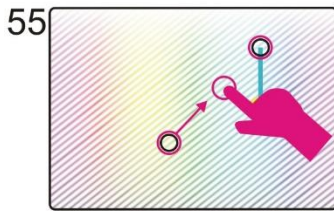
Bu seçimden sonra eller ekrandan çekildiğinde renk kontrol noktası kaybolur, ancak bir ilişkilendirme başladığında görünür hale gelecek



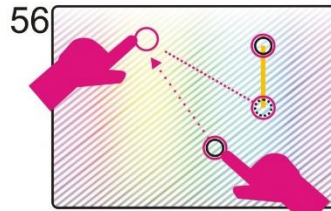
İlişkilendireceğimiz bir noktayı aşama 4-6'daki gibi yarattıktan sonra Aşama 20'deki gibi ilişkilendirme başladığında renk noktası görünür hale gelecektir



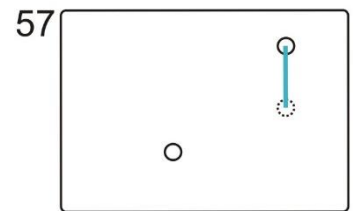
Noktanın konumu, bağlı noktalar nesnesinin rengi parametresine bağlanır



Seçim sonra



İlişkilendireceğimiz bir noktayı aşama 4-6'daki gibi yarattıktan sonra Aşama 20'deki gibi ilişkilendirme başladığında renk noktası görünür hale gelecektir

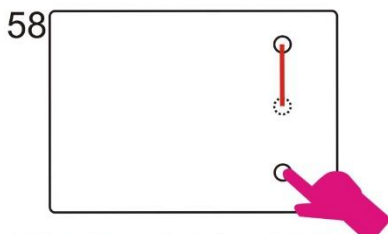


Eller ekrandan çekildiğinde renk altlığı ve rengi belirlemek için kullandığımız nokta kaybolur

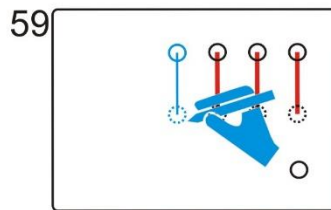
Ek bilgi :
Nesnelerin niteliklerinin kontrolü a n g e Ö:

Öneri yazılımda, bazı nesnelerin renk çizgi tipi gibi değişkenleri vardır. Bu değişkenler nesne yaratıldığında kontrol edilemez. Ancak nesne yaratıldıktan sonra bu değişkeni kontrol eden bir noktanın o nesneye ilişik olarak yaratılması gerekir. Bu kontrol noktasının ekrandaki pozisyonu, ilişik nesnenin rengi, çizgi tipi vb niteliklerini değiştirir. Bu kontrol noktası aynı normal noktalar gibi başka noktalarla ilişkili hale getirilebilir, bu sayede nesnenin niteliklerini, o nesneden farklı diğer nesnelerin ekrandaki pozisyonlarıyla kontrol edilebilir hale getirir.

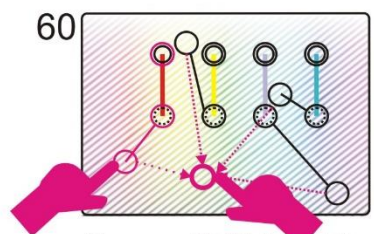
Bu nitelikler sadece renk, çizgi tipi değil (yukarıda ki bağlı nokta nesnesinde bunlardı), ileride açıklanacak daha farklı nesneler için daha farklı nitelikler olarak karşımıza çıkacaktır.



Artık noktanın ekranda yeri değiştirildiğinde, bağlı noktalar nesnesinin de rengi değişecektir.

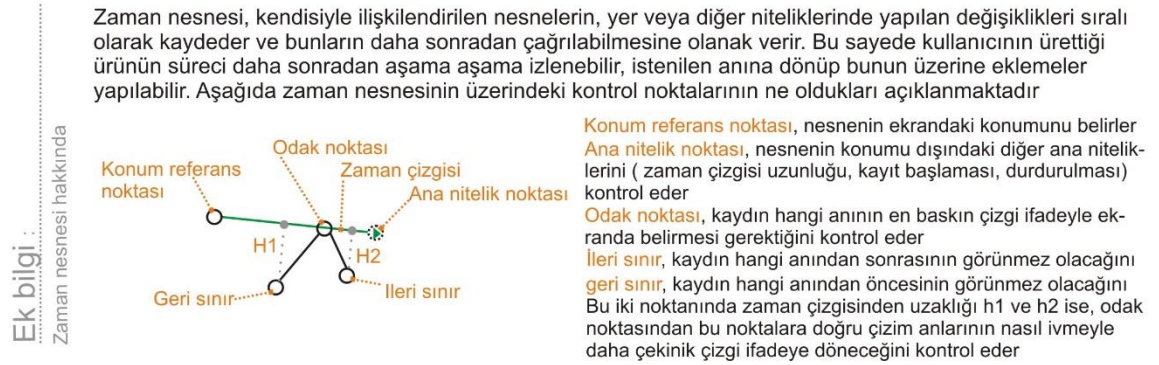
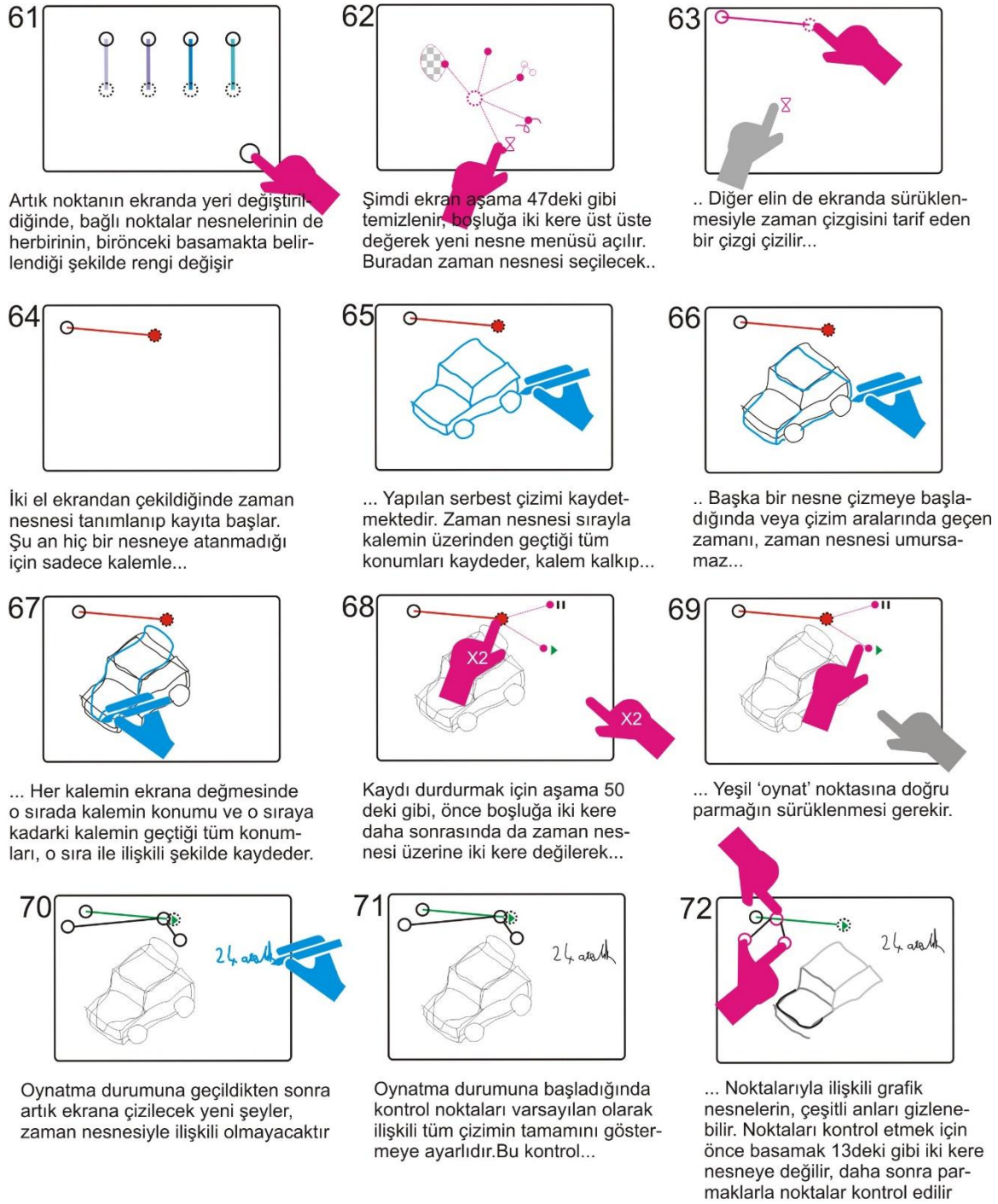


Şimdi bu nesnenin 4 kopyası çıkartılsın. Bunun için daha önceki aşama 38-41 arasında yapılan işlemlerin aynısıyla kopyalanır..

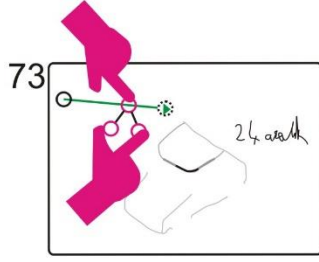


.. Ve aşama 55-56 daki gibi her bir bağlı nokta nesnesinin renk kontrol noktası ekranın sol altındaki noktaya bağlanır

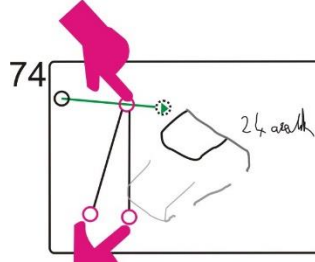
Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



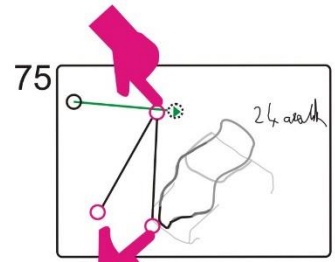
Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



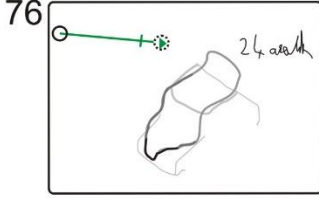
Bir el ile zaman çizgisi üzerindeki noktaya değil, diğer elin parmaklarıyla bu noktaya bağlı diğer noktalara değil, ve bunlar sürüklenir...



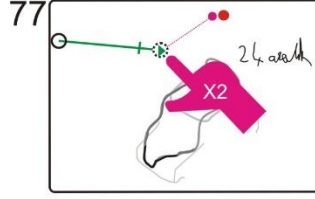
Zaman çizgisi üzerindeki nokta buradaki araç çiziminin hangi basamağının koyu renkle görüneceğini diğer sınır noktaları ise hangi...



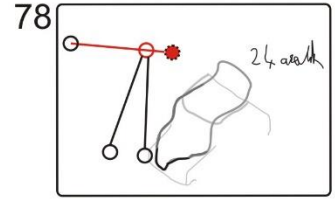
... Aşamalardan öncesi veya sonrası görünmez hale geleceğini gösterir, bu iki noktanın, zaman çizgisinden uzaklaşması veya...



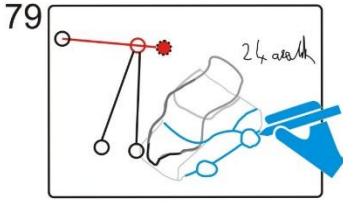
... Yakınlaşması ise odak noktasıyla sınır arasında çizilmiş çizgilerin nasıl bir ivmeyle silik hale geleceğini kontrol eder. Boşluğa parmakla değilip zaman nesnesi kontrolü sonlanır.



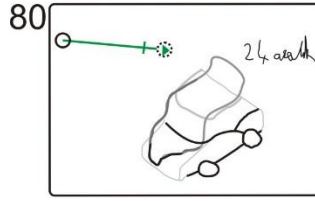
Şimdi ekrandaki çizime sadece ekrandaki anı için eklemeler yapalım. Bunun için iki kere nesne üzerine değilip çıkan menüden kayıt seçeneği üzerine parmak...



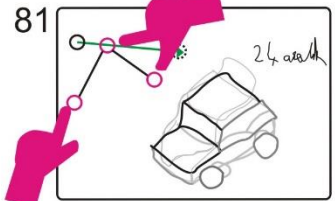
...Sürüklenerek kayıt modu açılır. Zaman nesnesi tekrar kırmızı renk almıştır ve üzerinde üç nokta vardır. Bu noktalar en son seçili odak, alt ve üst sınırları gösterir...



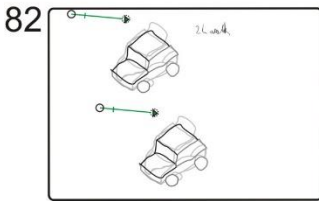
... Bu noktalar istenirse oynatılarak aynı aşama 73-75 deki gibi tekrar bir aralık ve odak 'an' seçimi sağlanır. Artık kalem çizdikleri buraya kayde-lice-lik-tilir.



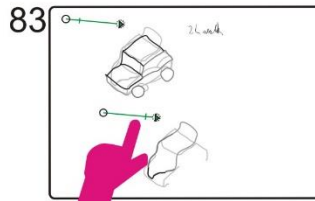
Aşama 68'dekileri yaparak tekrar oynatma durumuna geçildiğinde yeni yapılan ek, kayıt aralığında çizime eklenmiş olur.



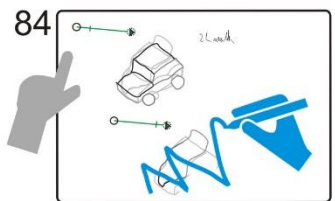
Aşama 72-75 deki gibi zaman nesnesiyle oynanıp çizimin çeşitli anları saklanır/vurgulanır. Yapılan ek çizim de eğer seçili aralık içindeyse ekranda görünür



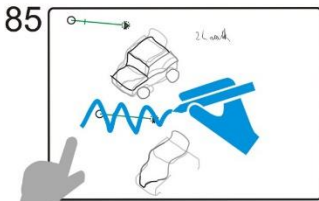
Aşama 35 deki gibi uzaklaşarak ve aşama 38 deki gibi zaman nesnesini seçerek kopyaladığımızda, onunla beraber ilişik araba çizimi de kopyalanır



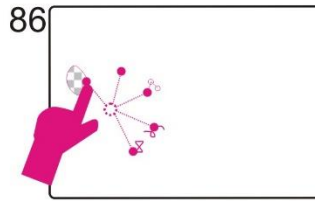
Aşama 73-75 deki gibi zaman nesnesiyle oynandığında, sadece ilgili çizim değişir, diğer çizim değişmez



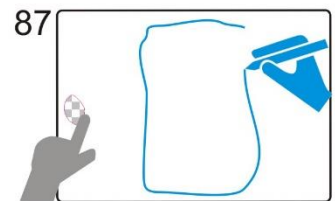
Şimdi ekrandaki araba çizimini silelim, eğer aşama 47 deki gibi arabanın üzerini karalanırsa araç silinmez, çünkü araba çizimi...



...Zaman nesnesine bağlıdır. Bu yüzden ilgili zaman nesnesi silinmelidir. Bu şekilde tüm ekrandakiler silinir



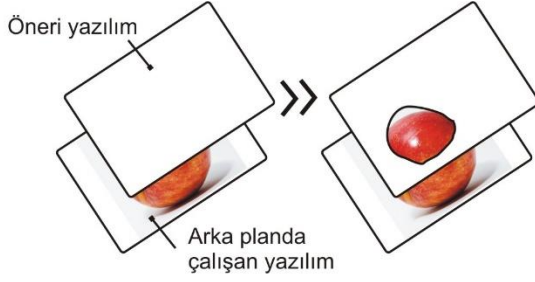
Şimdi aşama 3 deki gibi nesne menüsü açılıp yırtık nesnesine doğru parmak sürüklensin...



... Ve sonrasında parmak veya kalem ile ekranda bir alan çizilerek tanımlanır

Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo

Ek bilgi:
Yırtık nesnesi hakkında



Yırtık nesnesi, burada anlatılan yazılımın altında çalışan başka bir yazılımın, ekranda bir alan tanımlanarak, bu boşluktan görünmesini ve kontrol edilmesine izin veren bir nesnedir. Soldaki örnekte arka planda çalışan resim görüntüleme programında açık halde bırakılmış bir elma resmi, açılan yırtıktan görünmektedir. Ayrıca bu yırtık içinde parmak girdisi, sanki üstte çalışan bir program yokmuş gibi direk resim işleme programına gider.



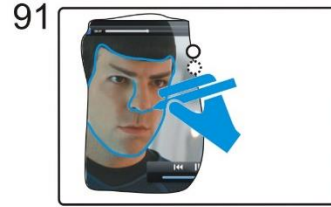
88 Bu alandan, bilgisayarda daha önceden açılmış arkada duran yazılım görünür



89 Şu an arkada bir video oynatıcı yazılımı açık. Parmakla yazılım ekranındaki tuşlara basılıp video oynatılabilir...



90 ... Veya yırtık nesnesinin konum referans noktasından aşama 14 deki gibi nesne sürüklenebilir



91 Yırtık üzerine kalemle çizim yapılabilir, bu girdi şekli parmakla değmek gibi arkada çalışan yazılıma ulaşmaz. Ayrıca her türlü diğer...



92 ... Grafik nesneler de yırtık üzerine konabilir, ve bunlara değerek yapılan girdi arkadaki yazılıma ulaşmaz



93 Aşama 86-87deki gibi yeni bir yırtık açıldığında yine arkada çalışan yazılım ekranı görülür



94 Bu sefer arkada çalışan video programı değil de web tarayıcı açılması istendiğinde iki kere yırtık nesnesi noktasına dokunulur...



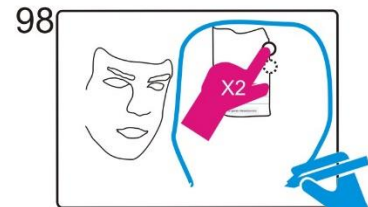
95 ... Parmak katmanlar imgesine sürüklendiği zaman o an arka planda açık tüm diğer yazılımlar görünür, bunlar arasından parmak google...



96 ...İmgesi üzerine sürüklendiğinde video oynatma yazılımı değil artık google tarayıcı açılır, fakat yırtık küçük olduğu için tam olarak...



97 ... Görünmüyor, bunun için çizilen resim aşama 47 deki gibi silinip...



98 ... Web tarayıcının görüntüğü yırtık nesnesine iki kere dokunulup tekrar kalemle bir alan tanımlandığında...



99 ... Yırtık nesnesi tekrar boyutlandırılmış olur.

Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo



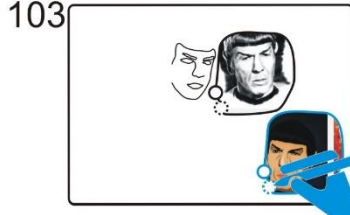
Yırtık nesnesinin içindeki google arama motoru, sanki üzerinde öneri bu yazılım yokmuş gibi çalışır. Şimdi el ile arama alanına değil kelime girdisi için klavye açılısın



Klavye ile bir anahtar kelime yazılıp anahtar kelimeyle ilgili imajlar çıksın. Bu senaryoda kullanıcı bir film karakteriyle ilgili bir çizim yapmaktadır.



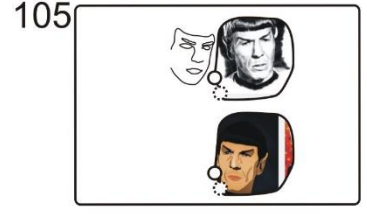
Bu karakterin internet üzerinde imajlarını bulup bunları kullanacaktır. Tablet ekranında boş yer açmak için aşama 35 deki gibi iki parmakla uzaklaşılır...



... Ve açılan yere aşama 38-41 deki gibi nesnenin bir kopyası konur. Yırtık nesnesi ekranda hareket ettikçe yırtıktan altında görünen kısımda değişir



Aşama 94deki girdi kombinasyonu ile açılan nitelik menüsünden o anki yırtıktan görünen kısım sabitlenebilir



Böylece nesnenin yeri değiştirilse bile yırtık altında gösterdiği kısım aynı kalır



Aşama 38deki girdi kombinasyonu ile yırtıktan görünen kısmın kitlenmediği üstte kalan yırtığa çok yaklaşırsa...



... Artık tamamen yırtık altındaki yazılım görünür. Sanki üstte başka bir yazılım çalışmıyormuş gibi alttaki yazılım kullanılabilir...



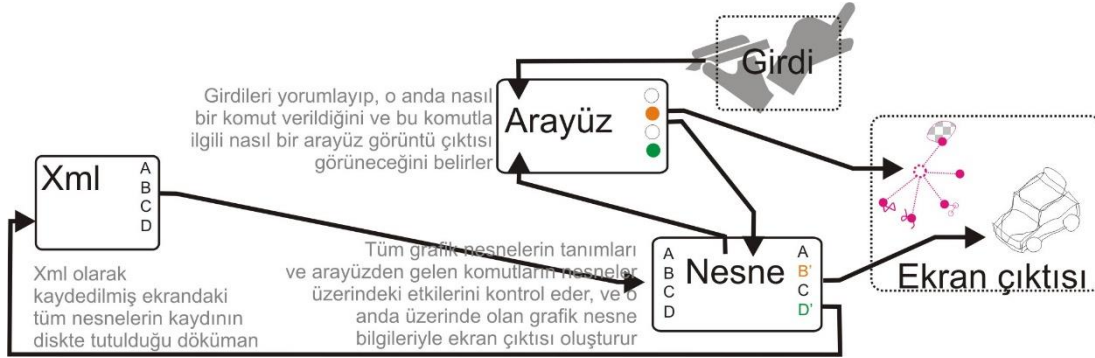
... Sadece kalem girdisi ve aşama 38 deki parmak girdisi kombinasyonu alttaki yazılımda çalışmaz. Bu girdi şekilleri üstteki çalışan öneri yazılımla ilişki kurar

Şekil 3.14 (Devamı) Modelin kullanım şeklini anlatır tablo

3.2.3 Modelinin yazılım kurgusu üzerinden anlatılması

Daha önce de bahsedildiği gibi, öneri model sadece kurgu olarak kalmış, bu modelin tamamen gerçekleştirildiği bir bilgisayar kodu yazılmamıştır. Fakat öneri modelde bahsi geçen bazı yazılım rutinlerinin kodları, modelin tecrübesinin nasıl olacağını görmek için yazılmıştır ve **Ek-1** de bu kodlar sunulmuştur. Bu kodlar, java-processing yazılım dilinde yazılmıştır ve ektede görünen birçok harici kütüphaneler kullanılmıştır. Bu yazılım parçaları tablette değil masaüstü bilgisayarda, girdi şekli olarak sadece klavye ve fare kullanarak çalışmaktadır.

Bu bölümde yazılım kodu kısmı düşülmüş bu rutinlerin/yazılım parçalarının ayrı ayrı nasıl kurgulandığı şekiller ve metinlerle açıklanmaktadır. Yazılımın genelinin de çalışma şekli basit olarak tarif edilmesi gerekirse; yazılım gözün anlayamayacağı kadar sık aralıklarla, devamlı güncellenen bir grafik nesneler kümesindeki kodları, o anda aldığı girdilerle işleyip, bu kümedeki nesnelerde gerekli ekleme değişikliği yapıp ekrana yazdırmaktadır (Şekil 3.15)



Şekil 3.15 Öneri yazılımın özet çalışma şekli diyagramı

İleriki alt başlıklarda 3.2.2 bölümünde sıralı resimlerde bahsi geçen çeşitli grafik nesneleri, aşama 21-63 arasında bahsi geçen ilişkili hareket, aşama 64-85 arasında bahsi geçen zaman nesnesi ve dokunulduğunda beliren arayüzün biçimlenişinin nasıl tanımlandığı hakkında çeşitli metin ve şekillerle açıklamalar bulunmaktadır. Bu sınıf, nesne ve fonksiyon tanımlamaları, bilgisayar yazılım bilgisi konusunda başlangıç düzeyinde olan kendim tarafımdan yapıldığı için, sadece yol gösterici ve şu anki durumu resmedici niteliktedir. Zira bu tanımlar işin ehli tarafından farklı şekillerde, daha verimli ve düzgün çalışacak şekilde tekrar yapılabilir. Bu yüzden bu kısımdaki bilgiler sadece nasıl bir yöntemin hedeflendiğini okuyanın da anlayabilmesi için yazılan iptidai uygulama fikirleridir.

3.2.3.1 Grafik nesnelerin tanımlanışı

Önerilen yazılımda her grafik nesne aşağıda listelenen değişkenlerle tanımlanır;

- Nesne ismi : Programda yaratılan her grafik nesne, sadece kendisine ait bir sayıyla tanımlanır.

- Nesne türü : Grafik nesnenin türünün bir sayı olarak tanımlandığı değişkendir. Sayının tanımlı olduğu nesne türünün tarifi yazılım kodunun için tanımlanmış, bu değişken yalnızca bu tanımlardan hangisine ait olduğunu tanımlayan bir sayıdır (örneğin 0 referansız serbest çizim, 1 nokta, 2 ilişkili nokta, 3 referanslı serbest çizim, 4 zaman, 5 yırtık nesnesi) .

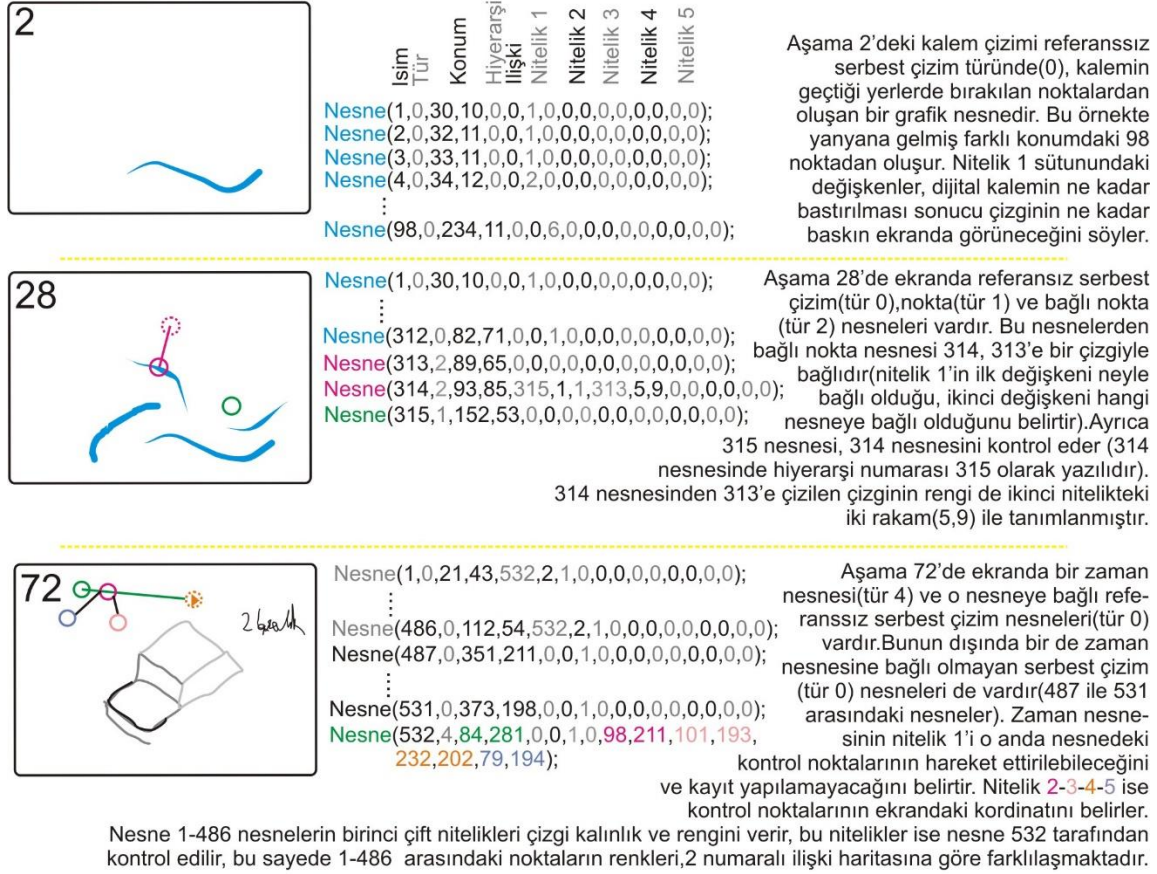
- Konum : grafik nesnenin o esnada ekrandaki x ve y eksenindeki konumunu belirten iki sayıdır.

- Hiyerarşi : Eğer nesne başka bir nesneyle ilişkiliyse ve bu ilişkide kontrol edilirse hangi nesne tarafından kontrol edildiğini belirtir. Eğer bir nesne başka bir nesne tarafından kontrol ediliyorsa onun konum ve aşağıda diğer niteliklerle bahsi geçecek nitelikleri kitlenip kullanıcı tarafından ulaşılamaz, bunlar sadece bu nesneyi kontrol eden diğer nesneden gelen bilgiye göre atanır. Hiyerarşi niteliği sayıyla tanımlanır. Bu sayı 0 ise nesne kontrol altında değil, bundan farklı bir rakam ise de o rakama karşılık gelen isimdeki nesnenin kontrolü altındadır.

-İlişki sıra numarası : Birbiriyle ilişkili nesneler varsa ve bu ilişki de bu nesne edilgen konumdaysa, bu ilişkinin haritalandığı ilişki sınıfı nesnenin isim numarasıdır (ilişki haritalanma methodu için bölüm 3.2.3.4'e bakınız). Bir sayıyla tanımlanır. İlgili ilişki nesnesi içindeki fonksiyonun sonucuna göre bu nesnenin yer veya renk gibi nitelikleri belirlenir)

-Diğer nitelikler : her nesnenin, nesne türü içinde tanımlanan birçok farklı niteliği vardır. Bu nitelikler çeşitli sayı veya sayı çiftleriyle ifade edilirler. (örneğin bağlı nokta nesnesinde renk veya zaman nesnesinde kontrol noktalarının konumları)

Yukarıda anlatılan grafik nesne niteliklerinin, grafik nesneleri nasıl tanımladığının anlaşılması için, 3.2.2 bölümündeki ekran görüntülerinden birkaçı, karşısında içerdiği nesnelerin yazılım tanımlarıyla beraber gösterilecektir (**Şekil 3.16**). Bu şekillerde görüldüğü gibi nesnelerin isim, konum gibi her bir nesne için aynı şeyleri ifade eden değişkenleri olduğu gibi, her nesnenin kendine has değişkenleri de kodun en sondaki numaralarıyla tanımlanır.



Şekil 3.16 Grafik nesne tanımları ve görüntüleri

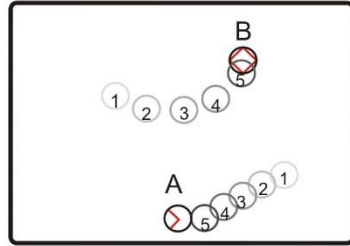
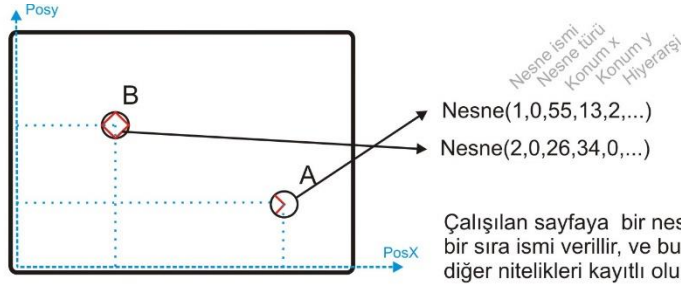
3.2.3.2 İlişkili hareket kaydı

Yazılım önerisinde her grafik nesnenin ekranda bir konumu vardır. Nesnelerin konumları yaratılmalarından sonra direkt parmakla nesnenin üzerine dokunup sürükleyerek değiştirilebilir. İki veya daha fazla nesnenin yeri aynı anda, farklı parmaklarla bu nesnelere dokunup sürüklenerek de değiştirilebilir. İki maddi nesnenin yerlerini beraber değiştirme süreci yarattığı sonuç (örneğin maketi ayrı yapılmış iki binanın yerleşimine karar verirken bu maketleri bir altlık üzerinde değişik yerlere koyarak karar vermek gibi) , bu iki nesnenin birbirine göre değişik durumlarda nasıl durmasının kullanıcının aklına göre doğru olacağını gösteren bir olasılık-cevap kümesidir. Bu karar verme/irdeleme yönteminin, iki veya daha fazla nesnenin aynı anda hareket ettirilmesine izin veren öneri modelde potansiyelinin daha iyi kullanılabilmesi için, yapılan bu hareketin kaydedilip yeniden oynatılabilmesi düşünülmüştür.

Fakat bu kaydetme ve yeniden oynatma bir video kaydı gibi değil, iki nesneden birinin hareket ettirilmesi sonucu diğer nesnenin, kayıt esnasında iki nesnenin yaptığı hareketleri göz önüne alarak, diğer nesneyi oynatması yoluyla yapılır.

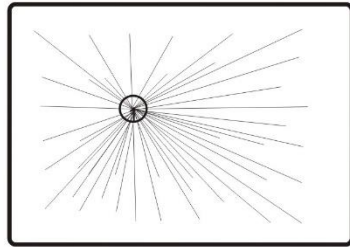
İlişkili hareket için ilk önce kontrol eden ve kontrol edilen nesnelerin kullanıcı tarafından belirlenmesi gerekir. Bu atama yapıldıktan sonra artık kontrol eden nesnenin ekrandaki konum değişkenlerindeki (x ve y kordinat değerleri) değişim, kontrol edilen nesnenin hangi değişkenine atanmışsa (konum değeri, renk değeri vb. olabilir) o değişkeni kontrol eder. Bu kontrolü ise bir karşılaştırma tablosuna bakarak yapar. Örnek vermek gerekirse 1 nesnesi kontrol eden nesne olsun, 2 nesnesi de kontrol edilen nesne olsun. 1 nesnesinin hareketiyle x koordinat değerindeki değişim, 2 nesnesinin x koordinat ekseninde ne kadar hareket ettireceğini, yazılım bu karşılaştırma tablosuna bakıp yapar. Bu yüzden her bir ilişki tarif edilirken, bu karşılaştırma tablosu da programa tarif edilir. Kullanıcı bu tarifi 1 ve 2 nesnesini ikisine aynı anda hareket ettirme yoluyla yapar. Yazılım 1 nesnesinin hareket esnasında geçtiği her konumun x ve y değerinin karşısına 2 nesnesinin o anda geçtiği konumun x ve y değerini koyar. Kullanıcı bu nesnelerin ekran üzerindeki hareketini bitirip ellerini nesneler üzerinden çektikten sonra, artık 1 nesnesi tekil olarak ekranda hareket ettirildiğinde, yazılım karşılaştırma tablosunda 1 nesnesinin o anki konumuyla, kayıt esnasında geçtiği en yakın noktayı tespit eder, daha sonra da bu kayıt esnasında geçtiği noktalardan en yakınındakinin konumundayken 2 nesnesinin hangi konumda olduğunu karşılaştırma tablosundan bulur ve 2 nesnesini oraya taşır.

Bu işlem, kullanıcının kayıt anında ekranın bütün noktalarını tarayan bir kayıt yapmaması, nesnelere çoğu zaman sadece lokal hareketler yaptırması, bu karşılaştırma tablosunda birçok koordinat hakkında hiçbir girdi olmamasına sebep olur. Bu kayıt sonrasında kontrol eden nesnede yapılacak hareketin, kayıta karşılaştırma tablosuna işlenen bir hareketin aynısı olmaması durumunda, kontrol edilen nesnenin istenmeyen şekillerde hareket etmesine neden olacaktır. Bu durumu yumuşatmak için **Şekil 3.17**'de de açıklandığı gibi bir yuvarlama yöntemine gidilmiştir. Bu tabloda aynı zamanda yukarıda anlatılan ilişkinin karşılaştırma tablosuna işlenmesi ve çağırılması süreci de açıklanmaktadır.



Kayit(0)[1] = nesne(1,0,55,13,2,...), Node(2,0,26,34,0,...)
 Kayit(0)[2] = nesne(1,0,50,10,2,...), Node(2,0,32,30,0,...)
 Kayit(0)[3] = nesne(1,0,46,8,2,...), Node(2,0,38,30,0,...)
 Kayit(0)[4] = nesne(1,0,42,6,2,...), Node(2,0,44,33,0,...)
 Kayit(0)[5] = nesne(1,0,39,3,2,...), Node(2,0,49,39,0,...)

Eğer kayıt işlemi başlatılırsa, bir kayıt nesnesi yaratılır ve içine ekrandaki tüm nesnelerin nitelikleri kaydedilir. Ekrandaki nesnelerin her hareketi için nesne içindeki listede yeni bir satır açılır, bu sayede sırayla nesnelerin tüm anları kaydedilir

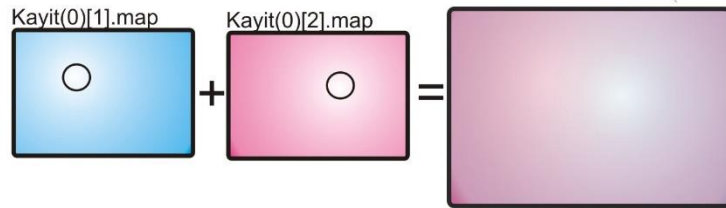


```
Mapkayit(kayit(0)){
  for(x=0;x<width;x++){
    for(y=0;y<height;y++){
      for(i=0;i<kayit(0).length;i++){
        kayit.map[i,x,y].distance=dist (kayit(1)[i].posx,kayit(1)[i].posy,x,y);
      }
    }
  }
}
```

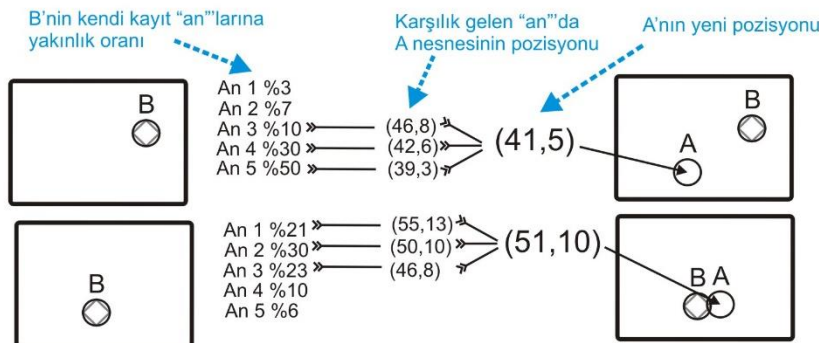
Kayıt işlemi sonlandırıldıktan sonra, kayıt nesnesi kayıt yapmayı bırakıp mevcut kayıtlarını haritalandırır. Haritalandırma her bir kayıt anı için ekrandaki nesnelerin konumları ile ekrandaki her bir yer için (bütün x,y düzlemi) uzaklık hesaplama işlemi yapar

Record(0)[1].map

20	12	8	12	20	25	28
13	7	4	7	13	18	23
13	7	4	7	13	18	23
20	12	8	12	20	25	28
25	19	12	19	25	29	34



Böylece ekranın hangi noktasının, her bir anda o noktaya ne kadar uzaklık olduğunu anlatan bir haritaya sahip olunur. Her hareket anı birbiri üzerine düşürüldüğünde, hangi ekranın hangi boşluğunun, hareket ettirilen noktaların her hareket anındaki konumlarına ne kadar uzakta olduğunu anlatan bir harita ortaya çıkar

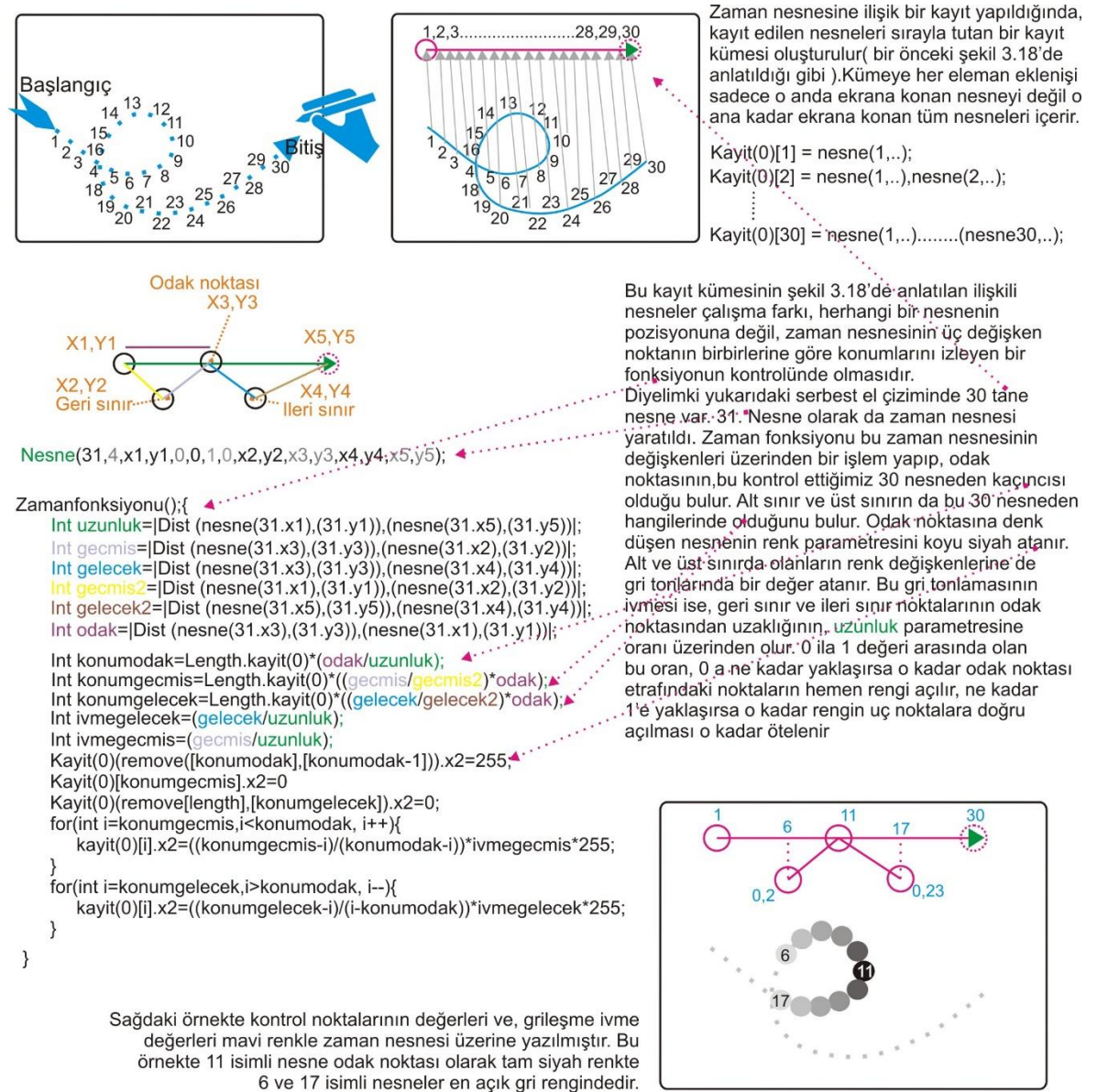


Kayıt tamamlandıktan sonra b noktasının yerinde yapılacak değişiklik sonrasında b nin yeni konumunda haritadaki verilere bakılır. Bu verilerden en yüksek oranlı 3 an aralarında çekilip kendi aralarında tekrar oranlanır. Bu oranla bu üç anda A'nın üç konumu arasında da bir oran kurulup, ortalama bir kordinat belirlenir. B'deki yapılan yer değişikliği sonucu A'nın kaydığı nokta, bu hesap sonucu çıkan noktadır

Şekil 3.17 Grafik nesnelerin birbirleriyle, konumlarıyla tanımlanan ilişkileri

3.2.3.3 Zaman Nesnesi

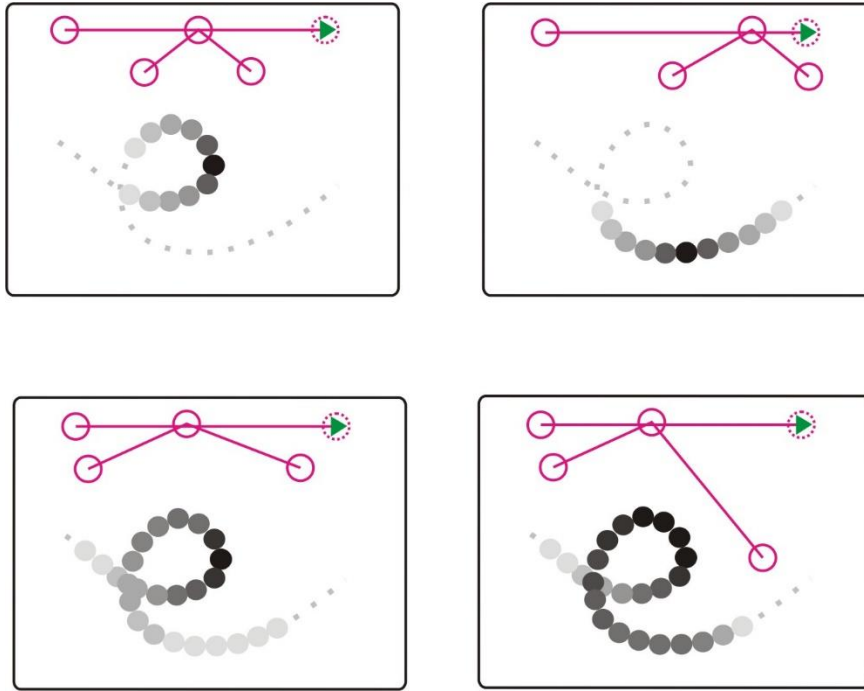
Zaman nesnesi basitçe tariflenmesi gerekirse, bugün birçok çizim programındaki *history log*'a benzer, grafik nesneler üzerinde yapılan değişiklikleri sırayla kaydedip bunların tekrar çağırılmasını sağlayan bir araçtır. Fakat bu yaygın araçlardan farklı olarak sadece sondan başa doğru yapılan değişikliklerin geri alınması şeklinde değil, şekil 3.18'de de tarif edildiği gibi, baştan yapılan şeylerin kaybedilebilmesi, istenilen zaman aralığının görülmesi, sadece istenilen anın görülebilmesi veya istenilen anın öncesi ve sonrasının görülebilmesi gibi niteliklerin kullanıcı tarafından kontrol edilmesine izin verebilecek şekilde düşünülmüştür.



Şekil 3.18 Grafik nesnelerin içinde tanımlanan başka nesneler

Zaman nesnesi bir önceki sayfadaki şekilde de anlatıldığı gibi, kendine bağlı grafik nesneler kümesindeki değişiklikleri ve eklemeleri kaydeder, fakat bu kayıt kümesinin her yeni elemanı, o anki yapılan değişiklik veya ekleme değil, o ana kadar yapılan işlerin tamamıdır, diğer bir deyişle devamlı kümülatif bilgi kayda geçer. Bilgisayara ek yük getirecek böyle bir kayıt şeklinin sebebi, zaman nesnesi bağlı nesneye daha sonra arada ekleme yapabilme, ondan bir nesne eksiltme veya değiştirme gibi durumlarda, bu değişikliği geçmişi yok etmeyerek yapabilmesine izin verecek olmasıdır.

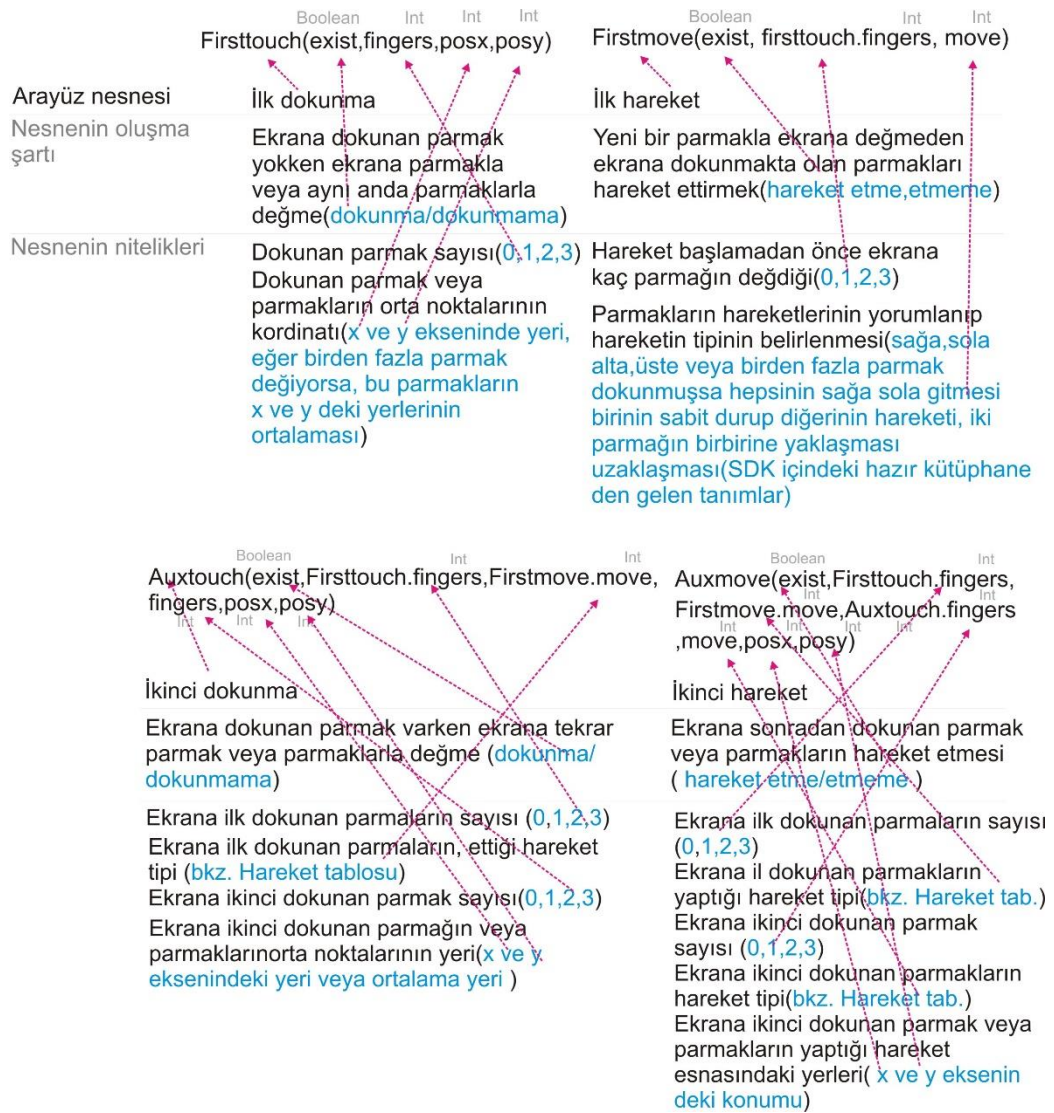
Şekil 3.18'in sonunda verilen zaman nesnesi ve ona bağlı referansız serbest çizim nesnesinin ekran görüntüsünün, zaman nesnesi parametresi değiştirildiğinde, çizimin geçmişte yapılan kısımları kontrollü şekilde kullanıcı tarafından saklanır/gözetilir. **Şekil 3.19**'da bunun nasıl sonuç verdiğini açıklamak için zaman nesnesinin değişkenlerinin çeşitli hallerinde çizimin nasıl görüldüğünü gösteren ekran görüntüleri mevcuttur.



Şekil 3.19 grafik nesnelerin içinde tanımlanan başka nesneler

3.2.3.4 Arayüz Nesnesi

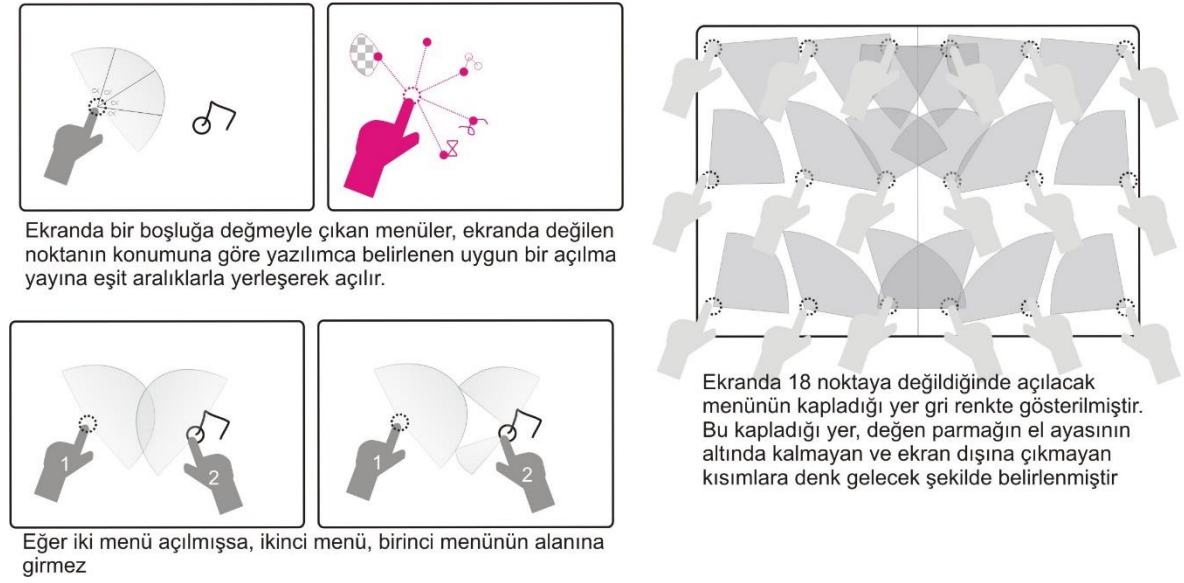
Arayüz nesnesi, parmak ve kalem ile tablet ekranı üzerine girilen konum bilgisini gözetler, ve bu gözetleme esnasında bunları sınıflandırır. Bu sınıflandırma yazılım geliştirme kitindeki hazır kütüphanedeki değme, eli çekme, sürüklenme, iki parmağın yaklaşması uzaklaşması hareket etmesi gibi tanımlı durumlara ek olarak, bu durumların çeşitli dizilimlerinden çeşitli komut yorumları da yapar. Yazılım parmak ve kalem konum verisinin çeşitli kombinasyonlarının ne kadar çeşitli girdi şekillerine izin verebileceği, bu kombinasyonların takibi için düşünülmüş kurgu kodun nesnelerinin anlatıldığı **Şekil 3.20** 'de görülebilir. Geleneksel tablet bilgisayar kullanma biçimi olan ekranda belli bir noktaya değme, onu geri bildirimini ekran çıktısı olarak görerek başka bir alana dokunma gibi, veya iki parmağın ekranda



Şekil 3.20 Parmak ve kalem konum girdisini gruplayan/yorumlayan nesneler

sürüklenmesi gibi bilindik kombinasyonlar dışında, program parmakların ekrana değış sırası gibi bilgilerin kombinasyonlarını da bir girdi şekli olarak yorumlar. Bu yorumlar da ekranda çizilen grafik nesnelerin kontrol edilmesini sağlar.

Bölüm 3.2.1’de anlatılan yazılım ilkelerinden biri “ekrana çizilen grafik nesneler dışında yazılım arayüzün çağırılmadığı zaman kesinlikle gözükmemesi, ekranda bir bölümlendirme olmaması ” için arayüz, ekranda normal kullanımda hiç görünmeyen ve ekranın her tarafında açılabilen bir liste olarak düşünülmüştür. Ekran üzerinde açılan seçenek listesinin açıldıktan sonra ekranda seçim bitinceye kadar açık kalması ve seçimin tekrar ekranda bir noktaya değilerek yapılması yerine, öneri arayüzde liste açıldıktan sonra parmağın bir yöne hareketiyle yapılan seçimler ve parmağın ekrandan çekilmesiyle kaybolan listeler düşünülmüştür. Ekranın herhangi bir yerine değildiğinde açılan bu liste kullanılabilir halde kalmak için, ekranda değdiği noktaya göre biçim değıştirir. Bu biçim değıştirme şekli **Şekil 3.21**’de anlatılan kurallara uyar.



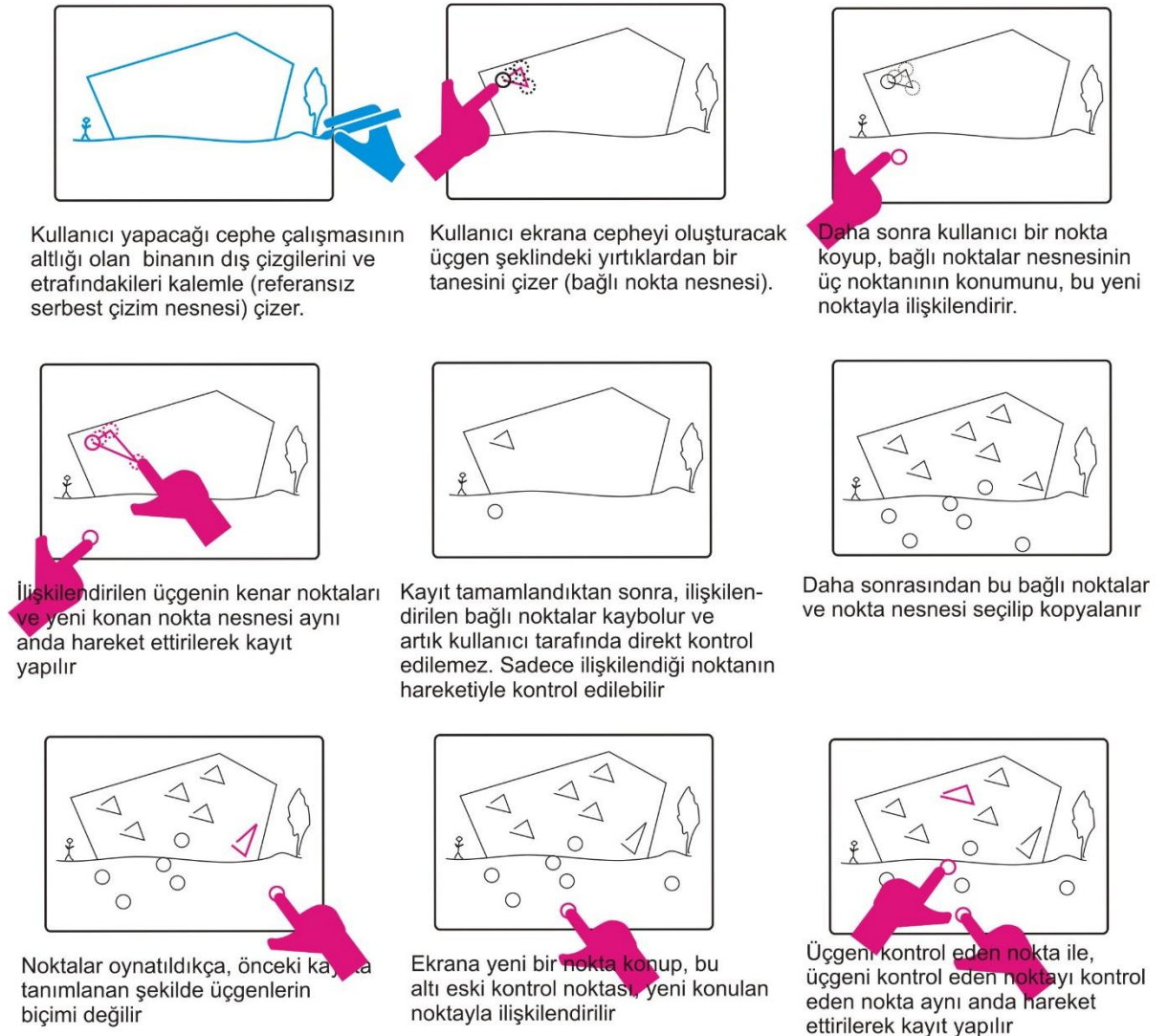
Şekil 3.21 Parmak ve kalem konum girdisini gruplayan/yorumlayan nesneler

3.3 Modelin Kullanım Örnekleri

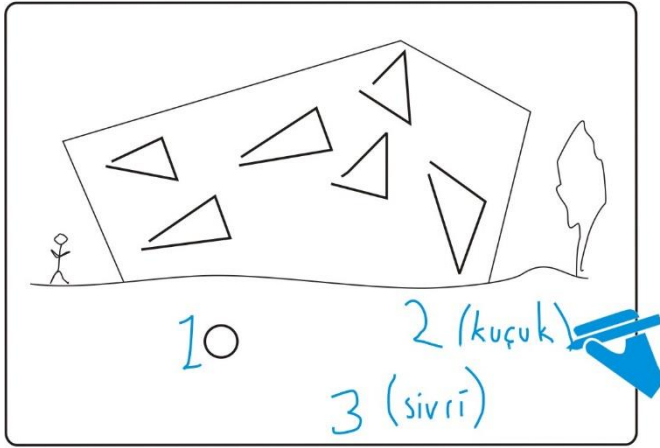
Bu bölümde bir tasarımcının yazılımı kullanma şeklinin anlatılacağı dört iş senaryosunu anlatan, altında açıklamalar olan kurgu sıralı ekran görüntüleri verilmektedir.

3.3.1 Eş Birçok parçadan oluşan cephe kompozisyonu

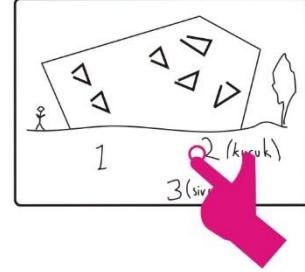
Bu senaryoda bir tasarımcı üçgen boşlukların kompozisyonundan oluşan bir bina cephesi denemesi yapmaktadır. **Şekil 3.22** 'de sıralı resimlerle kullanıcının yazılımı nasıl kullandığı kurgu ekran görüntüleriyle anlatılmaktadır. Kullanıcı cephenin altlığını çizdikten sonra ilk önce üçgen boşlukların nasıl değişik biçimlere dönüşebileceğini tanımlayacak, daha sonra bu üçgenleri ekrana dağıtarak çeşitli denemelerde bulunmakadır.



Şekil 3.22 Üçgen cephe kompozisyonu



Üçgenleri kontrol eden noktaların en son konan noktayla ilişki kayıtları tamamlandıktan sonra bu noktalar kaybolur. Artık bütün üçgenler bu noktanın hareketiyle kontrol edilir. Kullanıcı noktanın gittiği konumlarda çıkan sonuçları kendisi hatırlaması amacıyla notlar alabilir

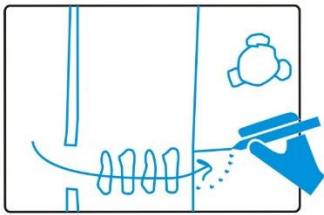


Şekil 3.22 (devamı) Üçgen cephe kompozisyonu

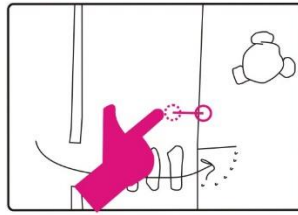
3.3.2 Bir kafeteryanın cephe elemanlarının plan düzleminde tasarlanması

Bu senaryoda kullanıcı bir kafeteryanın yola bakan cephesinde, mahremiyeti sağlayan düşey cephe elemanlarının biçimleri üzerinde denemeler yapmaktadır. Tasarımcı bahçe kapısından binaya girilirken içerinin görünmemesini istemekte, fakat bunu yaparken cephenin ritmini nasıl olacağı hakkında da deneme yapmak istemektedir.

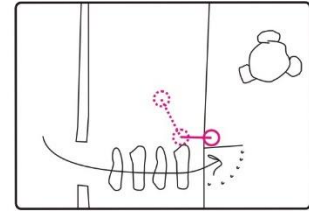
Şekil 3.23 'de anlatılan sıralı resimli kurgu ekran görüntülerinde kullanıcı mahremiyetin kaybolduğu durumları, cephe elemanlarının rengiyle ilişkilendirerek, cephe ritmini belirlerken mahremiyet hakkında geri besleme alabileceği bir sahne kurmaktadır.



Kullanıcı yapacağı cephe çalışmasının altlığı olan plan düzleminde binanın dış çizgilerini ve etrafındakileri kalemle (referansız serbest çizim nesnesi) çizer.

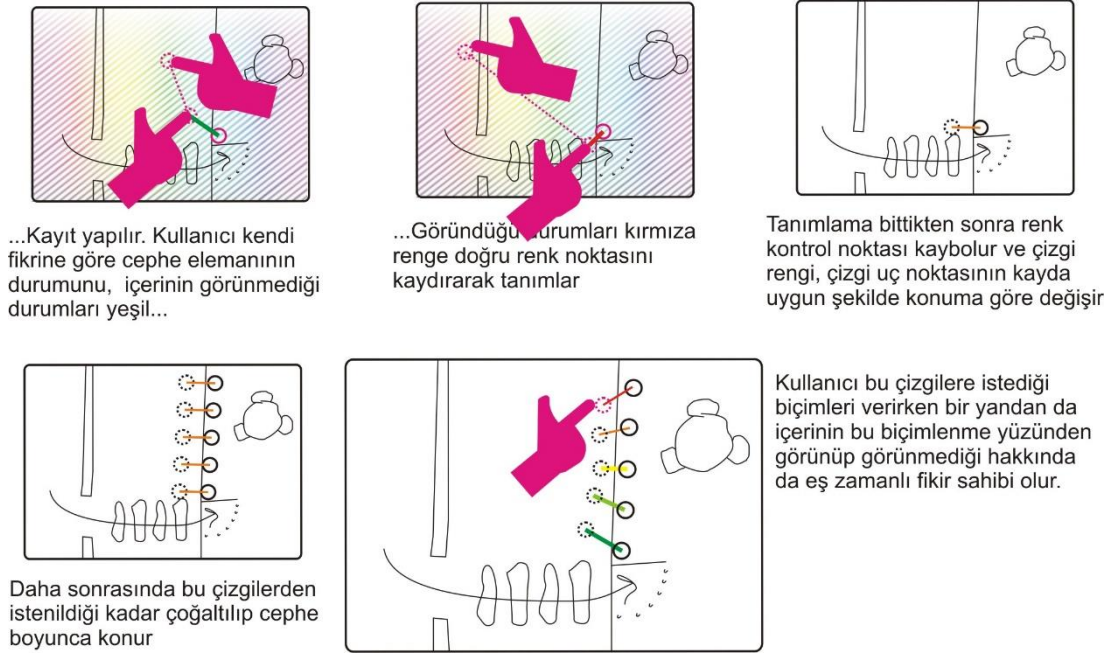


Kullanıcı ekrana cepheyi düşey cephe elemanını planda ifade edecek bir çizgi çizer (bağlı nokta nesnesi).



Bu çizginin renk değişkenini kontrol eden nokta yaratılır ve bu nokta uç noktasiyla ilişkilendirilip...

Şekil 3.23 (devamı arka sayfada) Kafeterya cephesi tasarımı



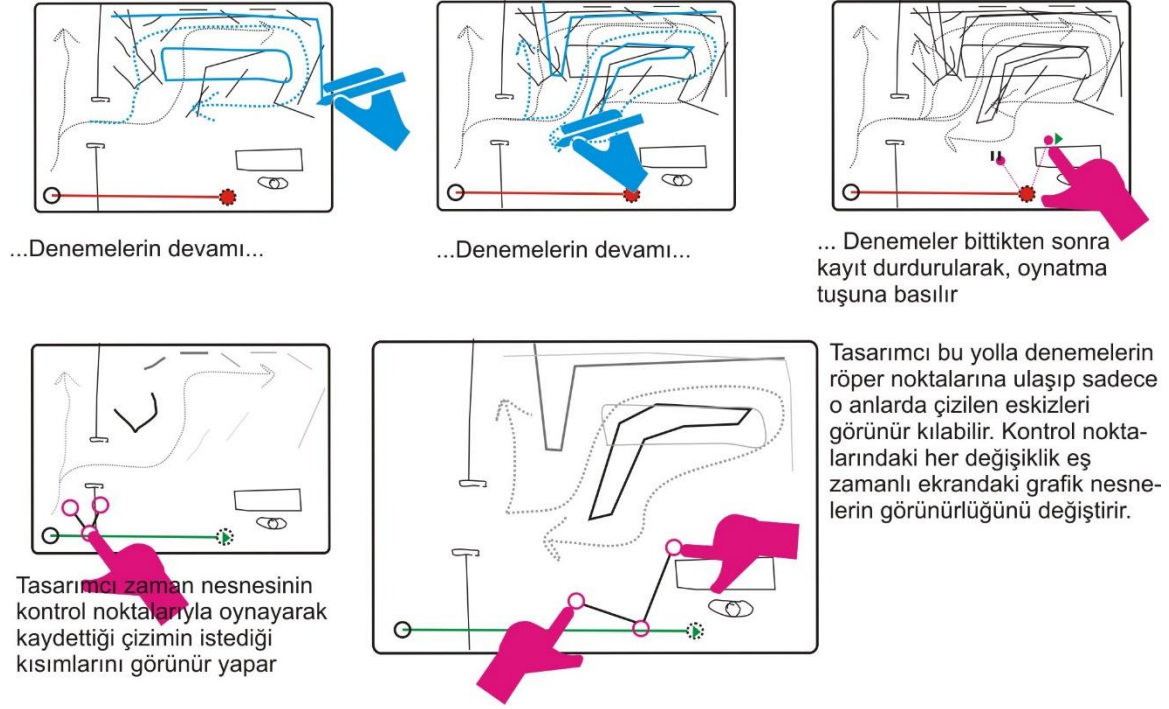
Şekil 3.23 (devamı) Kafeterya cephesi tasarımı

3.3.3 Plan düzleminde üst üste denemelerle tasarım

Bu senaryoda tasarımcı yazılımla serbest el çizim yaparak plan düzlemi üzerinde bir kitap dükkanı için vitrin düzeni tasarlamaktadır. Plan düzleminde vitrine ait seperatörleri, dükkan içindeki müşterinin çeşitli hareketleri üzerinden düşünme, bu hareketleri yönlendirme kararları hakkında denemeler yaparak tasarlamaktadır. Üst üste birçok insan hareketi ve seperatörlerin plan düzlemindeki biçimleri hakkında çizimler yapan tasarımcı, üst üste çizilen bu birkaç birbirinden ayrı durumu, zaman nesnesini kullanarak denemeleri bittikten sonra ayrı ayrı görebilmektedir (Şekil 3.24).



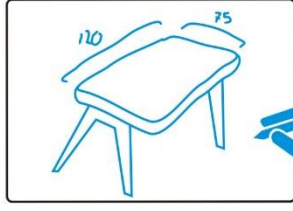
Şekil 3.24 (devamı) Üst üste denemelerle tasarım



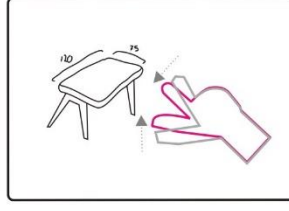
Şekil 3.24 (devamı) Üst üste denemelerle tasarım

3.3.4 Bir mobilya tasarımında malzeme seçimi yapılması

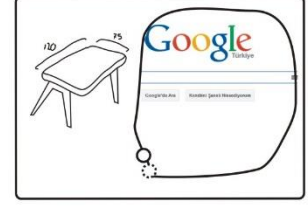
Bu senaryoda kullanıcı yazılımla serbest el çizim yaparak bir masa tasarlamış ve bunun kaplama malzemesine karar vermektedir. Bu kaplama malzemesine ait resmi internetten çağırıp, masayla aynı şekilde bir çerçeve içine koyup, bir kolaj yapmıştır (Şekil 3.25). Kullanıcı daha sonrasında istediği zaman tekrar bu resmin kaynağına, bu çerçeve üzerinden ulaşabilir, imalatçı bilgilerini bir kenara kalemle not alabilir. Ayrıca yine internet üzerinden başka ahşap dokusu bulup, onu yine aynı şekilde kullanabilir.



Kullanıcı yapacağı masanın eskizi yapar (referansız serbest çizim nesnesi).



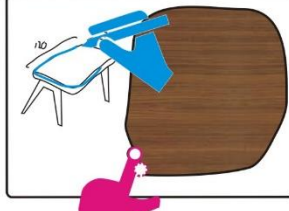
Kendine boşluk açmak için parmak hareketi yaparak çizdiği masadan uzaklaşır



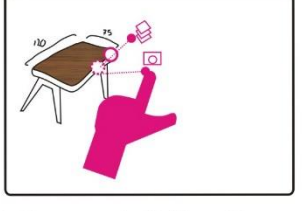
Daha sonra bir yırtık nesnesi açıp içinde web arama motorunda düşündüğü ahşap malzemenin görsellerini arar



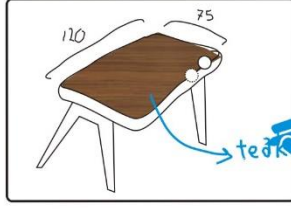
Düşündüğü malzemeyi seçer...



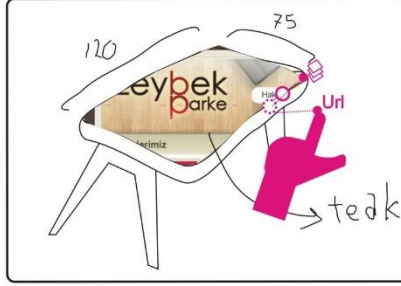
Daha sonra nesne tuşuna basılı tutarak yırtık nesnesini masanın üzerinde çizerek tekrar oluşturur



Yırtık nesnesi nitelik noktasına değilerek açılan menüden fotoğraf imgesine sürüklendiğinde yırtık içinde görüntü sabitlenir



Kullanıcı artık nesneye yaklaşıp da görüntü aynı kalır. Kullanıcı koyduğu bu ahşap dokusu üzerine çizim yapabilir



İstenildiği zaman yırtık nesnesinin nitelik noktasına değilerek bu imajın ait/bağlı olduğu web sitesine dönüş yapılabilir.

Şekil 3.25 Mobilya için malzeme seçimi

4. SONUÇ

Önerilen model yazılım, eskiz ile ilişkisel görsel içerik oluşturulan iş senaryoları için, kalem ile kağıda müdahalenin serbestliği ve kesintisizliğini olabildiğince kesmeden, bilgisayarın bu sürece nasıl yardımcı olabileceğini hakkında bir kurgu öne sürmüştür. Tez boyunca verilen örnekler genel olarak mimarın/tasarımcının fikrin erken aşamalarında eskizi kullanma şekilleri üzerinden örneklense de eskizi benzer şekilde kullanan her kişi için çeşitli kullanım potansiyelleri, öneri modelde mevcuttur. Yazılım kağıt kalem ve şekiller ile düşündüğümüz ve bunlarla kullanarak bir içerik, bir ilişkisel bilgi kümesi oluşturduğumuz işlerde kullanılabilir. Bu bir nesnenin tasarımı gibi somut bir iş olabileceği gibi, bir durumun çeşitli niteliklerinin biçime soyutlanarak kaydedilmesi ve üzerinde düşünülmesi gibi soyut bir alanlandaki bir kullanım senaryosu da olabilir.

Tüm tez boyunca örneklenen kullanımların ne kadar öngörüldüğü gibi işe yaradığını, ancak yazılımın çalışır halinin gerçek kullanıcılarla test edilmesi ile görülebilir. Bu test bu çalışmada yapılmamıştır. Modelin kurgu senaryolar üzerinden değerlendirilmesi gerekirse; model az aşamalı ve hassasiyet istemeyen tasarımın veya bir fikrin ilk aşamalarında, üzerindeki araçlarla yapılan işi kolaylaştırmakta ve kullanıcının çalıştığı hususların geçmişinin ve geçtiği aşamaların üçüncü bir kişi tarafından da okunabilmesini sağlamaktadır. Model sınırlıda olsa ilişkili grafik nesneler yapılabilmesi sayesinde kurallı biçim üretim aracı olarak da kullanılabilir. Bu biçimlerin kontrolü benzer kurallı biçim üretim programlarındaki tuşlar ve slider'lar yerine, bizzat onunla ilişkili biçimler veya kullanıcının yine serbest elle ürettiği bir kontrol biçimleriyle yapılmaktadır. Yani kurallı biçim ürettiği ve bu üretimi kontrol ettiği ortam aynı ortamdır. Bilgisayarda bugün, internet üzerindeki bir metin parçasını veya imajı, ancak aşamalı ve kurallı yollarla (resim, metin işleme vb. programlarıyla yapmak gibi) işlenebilmekte, bu yazılımların özelliklerinin gerekli olmadığı sadece bu bilgileri zahmetsiz ve aşamasızca yanyana koyma işinin önerilen modelde yapılabilmesi modelin pratikte en büyük artılarından birisidir.

Modelin kurgu senaryolar üzerinden değerlendirildiğinde zayıf yanlarından bahsetmek gerekirse; yazılım kurulan ilişkiler ve ekrandaki grafik nesneler arttıkça daha zor kullanılmaya başlanmaktadır. Bunun nedeni kullanıcı tarafından daha çabuk kullanılabilmek için tasarlanan arayüzün, daha geleneksel listeden çoktan seçme gibi arayüzlere göre birim ekran boşluğunda daha az seçenek sunmasıdır.

Model ayrıca hassasiyet gerektiren durumlarda da, bilgisayardaki yazılımlara göre çok daha yetersizdir. İşaretleme yolunun fare yerine parmak veya kalem gibi bir araç olması, numerik bilgi girilememesi, yapılan çizimin birbirlerinin noktalarını tutması veya düzgün çizgilerden oluşması gibi yardımcı araçların olmaması hassas çizimlerin yapılmasını tamamen engellemektedir.

Öneri yazılım ile oluşturulan ürünler sonuç ürün olmaktan ziyade, ara ürün olarak düşünülmelidir. Üretilen dijital eskizler, bundan sonra bilgisayarda başka yazılımla veya fiziksel olarak başka ortamda yapılacak iş için kullanıcı tarafından hazırlanan kılavuzlar olarak düşünülebilir. Yazılım kurgusunun kullanım senaryosundaki kullanıcı son ürün odaklı yapacağı işte, bu dijital eskizleri malzeme olarak kullanmaz, son ürünün yapılacağı yazılımda/ortamda sıfırdan işi oluşturmaya başlar. Zira fiziksel eskiz kullanımında da bu durum benzerdir.

4.1 Öneri Modeli Kuvvetlendirecek Teknolojiler

Öneri modelde ölçülü çizim yapılamaması ve düzgün çizgi çizilememesi, serbest el çizim hissiyatının kaybedilmemesi ve arayüzün karmaşıklıklaştırılmaması adına düşünülmüşse de, kağıt kalem ile yapılan eskizde bunların gerçekleşmesini sağlayan cetvelin, öneri modelde kullanılmayacağı anlamına gelmez. Ekranı parmakların hiç değmeyeceği bir dizilimle birkaç noktadan dokunacak bir cetvel, hatalı dokunma girdisine sebep olmadan yazılımla düz ve ölçülü çizgi çizilmesine olanak verebilir. Bugün tablet bilgisayarlar için buna benzer cetveller üretilmektedir (**Şekil 4.1**).

Bunun yanında donanımın kağıda daha fazla benzemesi adına bu yazılımın çalışacağı donanım, elektroforik ekrana sahip not defteri ölçülerinde, kısmen bükülebilen bir dijital not defteri olarak da kurgulanabilir (**Şekil 4.1**). Zira elektroforik ekranın arka aydınlatmaya ihtiyaç duymadan ortam ışığıyla çalışması, elektrik olmasa bile son ekran görüntüsünün devamlı ekran üzerinde kalması ve esnek olması, yazılımın eskiz metaforunu kuvvetlendirecek etmenlerdir (Url-6).



Şekil 4.1 Tablet üzerinde cetvel (Adobe digital ruler (Url5)) ve elektroforik ekran örnekleri (Url6)

Yazılımın önerdiği ilişkili bilgiyi oluştururken internet üzerindeki bilgiyi kullanma durumunun yarattığı önemli bir potansiyelden mevcuttur. Yazılımdaki yırtık nesnesi, içinden görülen arkaplan yazılımını dondurup son kullanıldığı ekrandaki parametrelerini kaydetmesi yüzünden, bir hypertext olarak nitelendirilebilir. Örnekle açıklamak gerekirse öneri yazılım modeli kapatıldığında, yırtık içinde açılmış web browserda yapılan aramalar, o anki açık web sayfası gibi veriler saklanır, ve elektronik eskiz tekrar açıldığında internetteki konumlarından bu bilgiler tekrar çağrılarak, yırtık içinde tekrar bu bilgiler görülür. Bu durumda öneri model ile yapılan kayıtlı her elektronik eskiz, dışarıdan çağırılan bilginin kaynağına bağlıdır. Eğer elektronik eskizler (mahremiyet sorunu yok sayılırsa) internet üzerinde saklanan ve herkesin ulaşabileceği dosyalar olursa, bu dosyaların içine çağırılan bilgiler üzerinden bunların tasnifi yapılabilir. Bu tersten tasnif bu toplu eskiz yığını, içinde kullanıldığı bilgiler üzerinden ulaşılabilen, ve o bilgilerin ne için kullanıldığını dışardan birisinin görebileceği, içinde gezinilebilen bir elektronik eskiz yığına çevirir.

Yazılımla oluşturulan eskizlerin internet üzerinde saklanması ve bu eskizler oluşturulurken kullanılan internetten çağırılmış bilgilerin, birbiriyle ilişkili şekilde arama motorlarında görülebilmesi, internet üzerinden yapılan aramalara yeni bir boyut kazandıracaktır. Şayet bu kullanım yaygınlaşırsa, arama motoru kullanıcıları aradıkları içeriklerin nasıl kullanıldıklarını görerek, çok farklı açılarda bu bilgilerin pratikteki kullanımını görmüş olacaklardır (Şekil 4.2)



Şekil 4.2 Kolaj, İnternet üzerinden bir kelime ile aratıldığında çıkan resimli sonuçların, bu resimleri internetten alıp kullanan elektronik eskizlerle beraber arama motorunda gözükmesi

KAYNAKLAR

- Apple**, (1996). Newton 2.0 User Interface Guidelines.
- Bush, V.** (1945). As We May Think, *essay for the Atlantic magazine*
- Davis, R.** (2007). Magic Paper: Sketch understanding Research.
- Davis, R., Alvarado, C.** (2001). Preserving the freedom of paper in a computer-based sketch tool.
- Davis, R., Alvarado, C.** (2000). Intelligent mechanical engineering design environment: From sketching to simulation.
- Goel, V.** (1995). Sketches of thought, M.I.T. Press.
- Key, A.** (1987). Doing With Images Makes Symbols: Communicating with Computers.
- Key, A.** (2006). Steps Toward the Reinvention of Programming: A compact and Practical Model of Personal Computing as a Self-exploratorium
- Oxman, R.** (2006). Digital architecture as a challenge for design pedagogy, Theort, Knowledge models and medium
- Oxman, R.** (2006). Theory and Design in the First Digital Age
- Purcell, Gero,** (1998). Drawings and the design process. Design Studies, Vol 19 (1998), pp. 389–430
- Prats, Garner,** (2006). Observations on Ambiguity in Design Sketches, Tracey, an internet-based drawing research journal, Loughborough University.
- Prats, Lim, Jowers** (1998). Transforming shape in design: observations from studies of sketching
- Schön, Wiggins** (1992). Kinds of seeing and their functions in designing. Design Studies, Vol 13 No 2 (1992), pp. 135–15
- Pedgley,** (2007). Capturing and analyzing own design activity
- Sutherland, I.** (1963). Sketchpad : A Man-Machine Graphical Communication System.
- Science Reporter,** (1964). Morach R.(Yönetmen). Computer Sketchpad [Televizyon Programı Bölüm İsmi] *Science reporter* [Televizyon Programı] içinde. U.S.National Educational Television.
- Url-1**<<http://en.wikipedia.org/wiki/Sketchpad>>, alındığı tarih: 06.04.2014.
- Url-2**<<http://www.db-rep.net/designtech/hci/sketching-dynamic-geometry/>>, alındığı tarih: 02.06.2013.
- Url-3**<<http://fluid.media.mit.edu/projects/glassified-intelligent-ruler-embedded-transparent-display>>, alındığı tarih: 02.02.2014.
- Url-4** <www.wacoom.com>, alındığı tarih: 11.12.2014.
- Url-5**<<http://www.adobe.com/products/ink-and-slide.html>>, alındığı tarih: 13.01.2015.
- Url -6** <<http://www.eink.com/flexible.html>>, alındığı tarih: 13.01.2015.

EK 1 – renk ilişkili nesne ve zaman nesnesinin tez sunumunda anlatımı yapıldığı processing ile derlenen yazılımın kodu.

```
PGraphics arkaplaneskiz;  
Nesne nesneler[]=new Nesne[0];  
Linememory linememories[]=new Linememory[0];  
int gecmisnoktax=50;  
int gecmisnoktay=27;  
int simdinoktax=250;  
int simdinoktay=21;  
int geleceknoktax=350;  
int geleceknoktay=16;  
float[] xposes;  
float[] yposes;  
float gecmisoran=0.3;  
float simdioran=0.6;  
float gelecekoran=0.8;  
float[] tx;  
float[] ty;  
float startx;  
float starty;  
int markednesne=0;  
boolean eskiz_dugme=false;  
boolean nesneeskiz_dugme=false;  
boolean nesnecizgi_dugme=false;  
boolean nesnezaman_dugme=false;  
boolean drag=false;  
boolean cizmeyeizin=true;  
boolean yarat=false;  
boolean cizgigrupbasi=false;  
void setup() {  
    strokeWeight(1);  
    ellipseMode(CENTER);  
    size(600, 600);
```

```

//smooth();
tx= new float[0];
ty= new float[0];
xposes= new float[0];
yposes= new float[0];
}
void draw() {
  if (dist(20, 60, mouseX, mouseY)<60)
    cizmeyeizin=false;
  else
    cizmeyeizin=true;
  background(250);
  if (nesnezaman_dugme==false) {
    arkaplaneskizgoster();
  }
  dugmeyaz();
  dugmekontrol();
  tumnesneleriyazdir();
  cagricizik();
  if (eskiz_dugme==true&&mousePressed==true) {
    arkaplaneskizyap();
  }
  if
(nesneeskiz_dugme==true&&mousePressed==true&&!drag&&cizmeyeizin&&!yara
t) {
    Nesne cereznesne=new Nesne(mouseX, mouseY, 1, false, 0, cizgigrupbasi);
    nesneler=(Nesne[])expand(nesneler, (nesneler.length+1));
    nesneler[(nesneler.length-1)]=cereznesne;
    yarat=true;
    cizgigrupbasi=false;
  }
  if
(nesnecizgi_dugme==true&&mousePressed==true&&!drag&&cizmeyeizin&&!yara
t) {
    Nesne cereznesne=new Nesne(mouseX, mouseY, 2, false, 1, cizgigrupbasi);
    nesneler=(Nesne[])expand(nesneler, (nesneler.length+1));

```

```

    nesneler[(nesneler.length-1)]=cereznesne;
    yarat=true;
    cizgigrupbasi=false;
}
hareketlidugme();
println(nesneler.length);
if (nesnezaman_dugme==true) {
    zaman();
}
}

void cagricizik() {
    if (keyPressed==true&&key=='a') {
        xposes=expand(xposes, xposes.length+1);
        xposes[xposes.length-1]=mouseX;

        yposes=expand(yposes, yposes.length+1);
        yposes[yposes.length-1]=mouseY;
    }

    if (xposes.length>1) {
        for (int i=1; i<xposes.length; i++) {

            line(xposes[i], yposes[i], xposes[i-1], yposes[i-1]);
        }
    }
}

void mousePressed() {
    if (nesnecizgi_dugme==false&&eskiz_dugme==false) {
        startx=mouseX;
        starty=mouseY;
        tx=expand(tx, xposes.length);
        ty=expand(ty, yposes.length);
        arrayCopy(xposes, tx);
        arrayCopy(yposes, ty);
    }
}

```

```

for (int i=0; i<nesneler.length; i++) {
    if (dist(mouseX, mouseY, nesneler[i].xpos, nesneler[i].ypos)<10) {
        nesneler[i].drag=true;
        drag=true;
    }
}
}

void mouseReleased() {
    for (int i=0; i<nesneler.length; i++) {
        nesneler[i].drag=false;
    }
    drag=false;
    yarat=false;
}

void mouseDragged() {
    if (nesnecizgi_dugme==false&&eskiz_dugme==false) {
        for (int i=0; i<tx.length; i++) {

            xposes[i]=tx[i]+mouseX-startx;
            yposes[i]=ty[i]+mouseY-starty;
        }
    }
}

void dugmeyaz() {
    noFill();
    if (eskiz_dugme==true) {
        stroke(120, 0, 244);
        //rect(0, 0, 40, 40);
        line(mouseX+100, mouseY-15, mouseX+108, mouseY-22);
        line(mouseX+108, mouseY-22, mouseX+110, mouseY-25);
        line(mouseX+110, mouseY-25, mouseX+124, mouseY-22);
        line(mouseX+124, mouseY-22, mouseX+130, mouseY-28);
    }
    if (eskiz_dugme==false) {

```

```

    stroke(0);
    //rect(0, 0, 40, 40);
}
if (nesneeskiz_dugme==true) {
    stroke(120, 0, 244);
    //rect(0, 40, 40, 40);
    ellipse(mouseX+100, mouseY-15, 10, 10);
    line(mouseX+100, mouseY-15, mouseX+108, mouseY-22);
    line(mouseX+108, mouseY-22, mouseX+110, mouseY-25);
    line(mouseX+110, mouseY-25, mouseX+124, mouseY-22);
    line(mouseX+124, mouseY-22, mouseX+130, mouseY-28);
}
if (nesneeskiz_dugme==false) {
    stroke(0);
    //rect(0, 40, 40, 40);
}
if (nesnecizgi_dugme==true) {
    stroke(120, 0, 244);
    //rect(0, 80, 40, 40);
    ellipse(mouseX+90, mouseY+40, 10, 10);
    line(mouseX+90, mouseY+40, mouseX+120, mouseY+15);
    ellipse(mouseX+120, mouseY+15, 10, 10);
}
if (nesnecizgi_dugme==false) {
    stroke(0);
    //rect(0, 80, 40, 40);
}
}

```

```

void dugmekontrol() {
    if (keyPressed==true) {
        if (key=='1') {
            if (eskiz_dugme==false) {
                eskiz_dugme=true;
            }
        }
    }
}

```



```

    } else {
        eskiz_dugme=false;
        cizgigrupbasi=true;
    }
}
if (key=='2') {
    if (nesneeskiz_dugme==false) {
        nesneeskiz_dugme=true;
    } else {
        nesneeskiz_dugme=false;
        cizgigrupbasi=true;
    }
}
if (key=='3') {
    if (nesnecizgi_dugme==false) {
        nesnecizgi_dugme=true;
    } else {
        nesnecizgi_dugme=false;
        cizgigrupbasi=true;
    }
}
if (key=='4') {
    if (nesnezaman_dugme==false) {
        nesnezaman_dugme=true;
    } else {
        nesnezaman_dugme=false;
        cizgigrupbasi=true;
    }
}
}

void hareketlidugme() {
    if (keyPressed==true&&key=='q') {
        strokeWeight(1);
    }
}

```

```

stroke(0);
ellipse(mouseX, mouseY, 20, 20);
line(mouseX, mouseY, mouseX+70, mouseY+30);
//
ellipse(mouseX+70, mouseY+30, 20, 20);
ellipse(mouseX+90, mouseY+40, 10, 10);
line(mouseX+90, mouseY+40, mouseX+120, mouseY+15);
ellipse(mouseX+120, mouseY+15, 10, 10);
//
line(mouseX, mouseY, mouseX+80, mouseY-30);
ellipse(mouseX+80, mouseY-30, 20, 20);
ellipse(mouseX+100, mouseY-15, 10, 10);
line(mouseX+100, mouseY-15, mouseX+108, mouseY-22);
line(mouseX+108, mouseY-22, mouseX+110, mouseY-25);
line(mouseX+110, mouseY-25, mouseX+124, mouseY-22);
line(mouseX+124, mouseY-22, mouseX+130, mouseY-28);
//
line(mouseX, mouseY, mouseX+30, mouseY-90);
ellipse(mouseX+30, mouseY-90, 20, 20);
line(mouseX+50, mouseY-75, mouseX+58, mouseY-78);
line(mouseX+58, mouseY-78, mouseX+60, mouseY-75);
line(mouseX+60, mouseY-75, mouseX+74, mouseY-78);
line(mouseX+74, mouseY-78, mouseX+80, mouseY-72);
strokeWeight(1);
}
}

void arkaplaneskizyap() {
    linememories=(Linememory[])expand(linememories, (linememories.length+1));
    linememories[(linememories.length-1)]=new Linememory(pmouseX, pmouseY,
mouseX, mouseY);
    line(pmouseX, pmouseY, mouseX, mouseY);
    for (int i=0;i<linememories.length;i++) {
        line(linememories[i].xpast, linememories[i].ypast, linememories[i].xnow,
linememories[i].ynow);
    }
}

```

```

}
void arkaplaneskizgoster() {
    for (int i=0;i<linememories.length;i++) {
        line(linememories[i].xpast,    linememories[i].ypast,    linememories[i].xnow,
linememories[i].ynow);
    }
}

class Nesne {

    float xpos, ypos, type;
    boolean  tikli, drag;
    boolean grupbasi;
    int cizgi;
    float renk=255;
    float renk2=0;
    Nesne(float x, float y, float t, boolean tik, int ciz, boolean grupbasi1) {
        xpos=x;
        ypos=y;
        type=t;
        tikli=tik;
        cizgi=ciz;
        drag=false;
        grupbasi=grupbasi1;
    }
    void nesneeskizyap() {
        stroke(120);
        ellipse(xpos, ypos, 30, 30);
    }
    void nesnecizgiyap() {
        stroke(0);
        ellipse(xpos, ypos, 20, 20);
    }
    void oncekinoktayacizgi(float oldx, float oldy) {
        stroke(255-renk-renk2, renk+renk2, 255);

```

```

    line(xpos, ypos, oldx, oldy);
}
void renkiliski(float iliskililposx, float iliskiliposy) {
    renk=dist(xpos, ypos, iliskililposx, iliskiliposy);
}
void renkiliski2(float iliskililposx, float iliskiliposy) {
    renk2=dist(xpos, ypos, iliskililposx, iliskiliposy);
}

void pozisyonguncelle() {
    if (drag) {
        xpos=mouseX;
        ypos=mouseY;
    }
}

class Nestedlinememory {
    int xnow, ynow, xpast, ypast;
    Nestedlinememory(int xp, int yp, int x, int y) {
        xnow=x;
        ynow=y;
        xpast=xp;
        ypast=yp;
    }
}

class Linememory {
    int xnow, ynow, xpast, ypast;
    Linememory(int xp, int yp, int x, int y) {
        xnow=x;
        ynow=y;
        xpast=xp;
        ypast=yp;
    }
}

```

```

void tumnesneleriyazdir() {
    for (int i=0;i<nesneler.length;i++) {
        nesneler[i].pozisyonguncelle();
        if ((nesneler.length)>2) {
            nesneler[2].renkiliski2(nesneler[1].xpos, nesneler[1].ypos);
        }
        if (nesneler[i].type==1) {
            nesneler[i].nesneeskizyap();
            markednesne=i;
        }
        if (nesneler[i].type==2) {
            if (markednesne!=0) {
                nesneler[i].renkiliski(nesneler[markednesne].xpos,
nesneler[markednesne].ypos);
            }

            nesneler[i].nesnecizgiyap();
        }
        if ((i>0)&&(nesneler[i-1].cizgi==1)&&(nesneler[i].cizgi==1)) {
            if (!nesneler[i].grupbasi)
                nesneler[i].oncekinoktayacizgi(nesneler[i-1].xpos, nesneler[i-1].ypos);
        }
        if ((i>0)&&(nesneler[i-1].cizgi==1)&&(nesneler[i].cizgi==1)) {
            if (!nesneler[i].grupbasi)
                nesneler[i].oncekinoktayacizgi(nesneler[i-1].xpos, nesneler[i-1].ypos);
        }
    }
}

void zaman() {
    strokeWeight(1);
    ellipse(30,30,20,20);
    line(30,30,400,15);
    ellipse(400,15,20,20);
    if(keyPressed==true&&key=='z'){

```

```

gecmisoran=((float)mouseX-30.0)/370.0;
gecmisnoktax=mouseX;
gecmisnoktay=mouseY;
}
if(keyPressed==true&&key=='x'){
    simdioran=((float)mouseX-30.0)/370.0;
    simdinoktax=mouseX;
    simdinoktay=mouseY;
}
if(keyPressed==true&&key=='c'){
    gelecekoran=((float)mouseX-30.0)/370.0;
    geleceknoktax=mouseX;
    geleceknoktay=mouseY;
}
ellipse(gecmisnoktax,gecmisnoktay,20,20);
line(gecmisnoktax,gecmisnoktay,simdioran,simdioran);
ellipse(simdioran,simdioran,20,20);
line(simdioran,simdioran,geleceknoktax,geleceknoktay);
ellipse(geleceknoktax,geleceknoktay,20,20);
fill(0,200,20);
triangle(393,7,393,23,410,15);
noFill();
for (int m=int (int (linememories.length)*(gecmisoran));
m<int(int(linememories.length)*(simdioran)); m++) {
    stroke(255-(255/(linememories.length*(simdioran-gecmisoran)))*(m-
int(int(linememories.length)*(gecmisoran))));
    line(linememories[m].xpast, linememories[m].ypast, linememories[m].xnow,
linememories[m].ynow);
}
for (int n=int (int (linememories.length)*(simdioran));
n<int(int(linememories.length)*(gelecekoran)); n++) {
    stroke((255/(linememories.length*(gelecekoran-simdioran)))*(n-
int(int(linememories.length)*(simdioran))));
    line(linememories[n].xpast, linememories[n].ypast, linememories[n].xnow,
linememories[n].ynow);
}
}

```


ÖZGEÇMİŞ

Ad Soyad : Turan ALTINTAŞ
Doğum Yeri ve Tarihi : Milas MUĞLA 1985
E-Posta : inbox@turanaltintas.info



ÖĞRENİM DURUMU:

- **Lisans** : 2008, İ.T.Ü., Mimarlık Fakültesi, Mimarlık Bölümü

MESLEKİ DENEYİM VE ÖDÜLLER:

- Savant İnternational, Proje Yönetimi ofisi. Mimar. 2008 Ağustos- 2009 Ocak. İstanbul.
- Kolart Mimarlık, Mimarlık ofisi.Mimar. 2011 Mart- 2011 Aralık.. AbuDhabi ve Muskat Uluslararası hava limanı ihaleleri için kontrol metraj setlerinin hazırlanması. İstanbul.
- Çarmıklı Saruhan Gay. İnş. Ortaklığı, Teknik Ofis ve Saha Mimarı. Ulus Savoy Evleri Şantiyesi. 2012 Nisan-2014 Şubat. İstanbul.
- Mimed 2005 öğrenci proje yarışması, 2.'lik . Öğrenci olarak.
- İ.M.P. 2006 Beylikdüzü Cum. Cad. Ulusal Mimari proje yarışması, 1.'lik . Yardımcı öğrenci olarak.
- İ.T.Ü. 2007 Taşkışla yanı arazi hakkındaki yarışma. 2.'lik ödülü. Yardımcı öğrenci olarak.
- İzmit Bel. 2010 İzmit Sahil Düzenlemesi Hakkında Yarışma, 1.'lik. Yardımcı mimar olarak.
- Zonguldak Bel. 2010 Kömür madeni liman tesisleri ve etrafının düzenlenmesi hakkında yarışma, 2. Mansiyon. Yardımcı mimar olarak.
- Afyon Bel. 2011 Afyon Cumhuriyet meydanı düzenlemesi hakkında yarışma, 2.'lik . Yardımcı mimar olarak
- Antalya 2014 Konyaaltı Sahil Düzenlemesi Yarışması, Mansiyon ödülü. Yardımcı mimar olarak

- Anamur 2014 Atatepe Sosyal Merkezi ve Çevresi Ulusal Mimari Proje Yarışması, Mansiyon ödülü. Yardımcı mimar olarak
- MEB 2014 Gökçeada lise kampüsü mimari proje yarışması, Mansiyon ödülü. Yardımcı mimar olarak