BİLGİSAYAR DESTEKLİ MOBİLYA TASARIMINDA AUTOLISP UYGULAMALARI

HAYRİ BULUT

YÜKSEK LİSANS TEZİ
MOBİLYA VE DEKORASYON EĞİTİMİ

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ

Temmuız 2010
ANKARA

Hayri BULUT tarafından hazırlanan "BİLGİSAYAR DESTEKLİ MOBİLYA TASARIMINDA AUTOLİSP UYGULAMALARI" adlı bu tezin Yüksek Lisans tezi olarak uygun olduğunu onaylarım.

Yrd. Doç. Dr. Nihat DÖNGEL ……..
Tez Danışmanı Mobilya ve Dekorasyon Eğitimi

Bu çalışma, jürimiz tarafından oy birliği ile Mobilya ve Dekorasyon Eğitimi Anabilim Dalında Yüksek Lisans tezi olarak kabul edilmiştir.

Yrd. Doç. Dr. Abdullah TOGAY ……..
Endüstriyel Teknoloji Eğitimi, Gazi Üniversitesi

Yrd. Doç. Dr. Nihat DÖNGEL …….
Mobilya ve Dekorasyon Eğitimi, Gazi Üniversitesi

Yrd. Doç. Dr. İhsan KÜRELİ …….
Mobilya ve Dekorasyon Eğitimi, Gazi Üniversitesi

Tarih: 02/07/2010

Bu tez ile G.Ü. Fen Bilimleri Enstitüsü Yönetim Kurulu Yüksek Lisans derecesini onamıştır.

Prof. Dr. Bilal TOKLU
Fen Bilimleri Enstitüsü Müdürü

TEZ BİLDİRİMİ

Tez içindeki bütün bilgilerin etik davranış ve akademik kurallar çerçevesinde elde edilerek sunulduğunu, ayrıca tez yazım kurallarına uygun olarak hazırlanan bu çalışmada bana ait olmayan her türlü ifade ve bilginin kaynağına eksiksiz atıf yapıldığını bildiririm.

Hayri BULUT

BİLGİSAYAR DESTEKLİ MOBİLYA TASARIMINDA AUTOLISP UYGULAMALARI
(Yüksek Lisans Tezi)

Hayri BULUT

GAZİ ÜNİVERSİTESİ
FEN BİLİMLERİ ENSTİTÜSÜ
Temmuz 2010

ÖZET

AutoLISP programlama dili lisp processing denilen lisp dilinin AutoCAD için uyarlanmış halidir. Lisp programlama dili yapay zeka (artificial intelligence) dilidir. Bu dilde AutoCAD fonksiyonları (çizim komutları vs...) ve kullanıcının kendine özgü geliştirebileceği fonksiyonları da kullanması mümkündür. AutoLISP programlama dilini AutoCAD'in desteklemesi, AutoCAD kullanıcılarına bu dili kendi konstrüksiyon problemlerinin çözümünde de kullanma imkanı sağlamaktadır. Bu çalışmada bilgisayar destekli mobilya tasarımında Autolisp kullanımına yönelik uygulamalar yapılmıştır. Bu maksatla mobilya tasarımında tasarımcının en fazla zaman harcadığı konstrüksiyon problemlerinin Autolisp ile çözümlenmesi hedeflenmiştir. Mobilya konstrüksiyonlarında yaygın olarak uygulanan kavelalı birleştirme ve zıvanalı birleştirme teknikleri örnek olarak seçilmişlerdir. Bu maksatla yazılan lisp programlarının nasıl çalıştığı örnekler üzerinde gösterilmiş, programların AutoCAD ortamında nasıl kullanılacağı anlatılmıştır. Çalışmada verilen örneklerde de görüldüğü gibi Autolisp tasarım süresini kısaltmakta, maliyeti azaltmakta rekabet gücünü artırmaktadır.

Bilim Kodu : **711.1.013**
Anahtar Kelimeler :**AutoCAD, Autolisp, Mobilya Tasarımı, Kavelalı birleştirme, Zıvanalı birleştirme**
Sayfa Adedi : **73**
Tez Yöneticisi : **Yrd. Doç. Dr. Nihat DÖNGEL**

THE USE OF AUTOLISP IN COMPUTER AIDED FURNITURE DESIGN
(M.Sc. Thesis)

Hayri BULUT

GAZİ UNIVERSITY
INSTITUTE OF SCIENCE AND TECHNOLOGY
July 2010

ABSTRACT

The AutoLISP is a language which called list processing and is used for AutoCAD. As it known, Lisp is an artificial intelligence language. In this language it is possible to use the functions of AutoCAD as well as the special subprograms developed by users. AutoLISP programming language that supports AutoCAD, AutoCAD users, this provides the possibility of using language to solve problems in their construction. In this study, computer-aided design of furniture made of applications for the use of AutoLISP. For this purpose, the most time spent by the designer's furniture design is aimed to resolve construction problems with AutoLISP. Widely applied in construction of furniture and grooved dowel and mortise and tenon joint techniques have been selected as an example. For this purpose, examples of how it works on the page shown in lisp programs, programs described how to use AutoCAD environment. As shown in given examples, the application of AutoLISP together with AutoCAD shortens the design time and improves the users competitiveness on the market.

Science Code : **711.1.013**
Key Words :**AutoCAD, Autolisp, Furniture Design, Dowel Joint, Mortise and Tenon Joint**
Page Number : **73**
Adviser : **Assist. Prof. Dr. Nihat DÖNGEL**

TEŞEKKÜR

Çalışmalarım süresince yardım ve katkılarıyla beni yönlendiren değerli hocam Yrd. Doç. Dr. Nihat DÖNGEL'e, Mobilya ve Dekorasyon Eğitimi Bölümü öğretim üyeleri, araştırma görevlileri ve personeline, program yazmada yardım ve desteğini esirgemeyen Mehmet GÜNEŞ'e teşekkürü bir borç bilirim.

İÇİNDEKİLER

Sayfa

Sayfa

ŞEKİLLERİN LİSTESİ

Şekil **Sayfa**

Şekil **Sayfa**

1. GİRİŞ

Bilgisayar ortamında düşünülen parçaların üç ya da iki boyutta tasarlanması ve modellenmesi, tasarlanan parçaların teknik çizimlerinin üretilmesi Bilgisayar Destekli Tasarım (CAD) işlemidir. Başka bir deyişle, Bilgisayar Destekli Tasarım (CAD), tasarımın yapılmasını kolaylaştırmak, hızlandırmak, kalitesini yükseltmek gibi amaçlara ulaşmak için araç olarak bilgisayardan yararlanma eylemidir. Genel olarak CAD üzerine beklentiler; üretkenlik, hassasiyet ve uyumluluk olarak özetlenebilir [1].

CAD sistemleri kullanımında ortaya çıkan büyüme çok geniş alanda ve farklı uygulama alanlarına sahip yazılım paketlerinin hızlı bir şekilde gelişmesini mümkün kılmıştır. Kullanıcı yazılımı aldığında, bu yazılımı çalıştırma sorumluluğunun kullanıcıda olması, ilk sistemlerin karakteristik özelliğidir. Bu durum son 30 yılda değişmiş ve 1970'li yılların başından itibaren kullanıcıyı yazılım problemlerinden kurtaran paket programlar üretilmiştir [2].

Türkiye orman ürünleri sanayiinin alt sektörleri arasında en büyük payı oluşturan ve önemli girdilerini bu sanayiden sağlayan mobilya işletmeleri, gelişmiş ülkelerin mobilya endüstrisindeki daralmanın aksine ülkemizde son yıllara dinamik bir gelişim sürecini yaşamaktadır [3, 4]. Ancak mobilya sanayiinin genel imalat sanayiine benzer şekilde, önemli işletmecilik problemleri bulunmaktadır. Orta ölçekli işletmelerin % 67'si işletme sermayesi ve teknolojik gelişmeleri izleme açısından finans sıkıntısı çekmektedir [5]. Özellikle küçük ve orta ölçekli işletmelerde ileri uygulama teknolojilerine pek rastlanmamaktadır. Sadece büyük ölçekli işletmelerde ve nadiren orta ölçekli işletmelerde üretimin belli aşamalarında ileri teknoloji uygulamalarına rastlanmaktadır [3].

Bilgisayar destekli tasarımın gözde olduğu günümüzde, Autolisp profesyonel AutoCAD kullanıcılarının olduğu kadar amatör kullanıcıların da ilgi duyduğu konulardan biridir. Ancak, birçok AutoCAD kullanıcısı Autolisp'ten nasıl faydalanacağını bilmemektedir [6].

AutoCAD programı tasarımcıya ilave programlar ve menüler üzerinde değişiklik yapma imkanı vermektedir. Bu özelliğinden dolayı ne kadar gelişmeye elverişli bir program olduğu anlaşılmaktadır. Kendi içinde mevcut bulunan özelliklerden bir çoğu yine kullanıcı tarafından ilave edilmiştir. Örneğin bir mimar kendi alanı ile ilgili çizim ve sembolleri üretip gerekli programları ilave ettikten sonra AutoCAD programı, mimari çizimler alanında daha verimli hale gelmiş olacaktır. Diğer yandan bir mobilya ve dekorasyoncu kendi alanına yönelik ilave LISP programları yazması AutoCAD'i bu alanda güçlü yapacaktır. Aynı şekilde tüm mühendislik alanlarında da uyarlamalar yapılabilir [7].

Bu çalışma ile mobilya ve dekorasyon alanında tasarım ve konstrüksiyonların Autolisp yardımıyla çözümlenerek AutoCAD ortamında kullanılmasına yönelik uygulamalar yapılmıştır. Bu sayede tasarımların daha kolay ve pratik olarak gerçekleştirilmesi için AutoCAD programının verimliliğini artırmak, çizimlerde sarf edilen zamanı azaltmak amaçlanmıştır [7].

2. GENEL BİLGİLER

2.1. Bilgisayar Destekli Tasarım

Bilgisayar Destekli Tasarım (CAD), tasarım işlemlerinin bilgisayar yardımı ile gerçekleştirilmesidir [8]. İngilizcesi Computer Aided Design olan Bilgisayar Destekli Tasarım, uluslararası platformda kısaca bu kelimelerin baş harflerinden oluşan CAD terimi ile anılmaktadır. Bilgisayar destekli tasarımda da önemli olan faktör insan faktörüdür. Her ne kadar bilgisayarı yönlendiren ve yöneten insan olsa da bilgisayar desteği ile grafiklerin oluşturulması, yüksek hız ile karmaşık tasarım analizlerinin yapılabilmesi, tasarımların saklanabilmesi ve yapılan işlemlerin saklanıp yeniden kullanılabilir olması çok daha rahat ve hızlı bir biçimde gerçekleştirilebilinmektedir. CAD, bir anlamda tasarım problemlerine bilgisayar uygulamalı çözümler olarak da yorumlanabilmektedir [9].

Literatürde CAD (Computer Aided Design) olarak adlandırılan bilgisayar destekli tasarım, genel bir tanımlama ile bir mühendislik tasarımının yaratılması, değiştirilmesi ya da dökuman haline getirilmesinde etkili bir bilgisayar kullanımı içeren tasarım faaliyeti olarak tanımlanır [10].

Bilgisayar destekli tasarım, geleneksel araç ve yöntemlere alternatif olarak ortaya çıkmış bir düşünce tarzıdır. Her türlü tasarım ve çizim konularında bilgisayardan yararlanmayı ve bilgisayarların desteklediği çizimleri içermektedir [10].

Bilgisayar destekli tasarım sayesinde, tasarımcılar yaptıkları tasarımın kontrolünü ve yaratacağı sonuçları bilgisayardan kısa sürede öğrenerek gerekli değişiklikleri hemen yapabilirler. Bu şekilde adım adım ilerlenerek en uygun tasarım elde edilir. Kontrol hesaplarının ve değişikliklerinin bilgisayar sayesinde kısa sürede yapılabilmesi, tasarım aşamasını daha verimli hale getirir [6].

CAD, bilgisayar teknolojisi ile mekanik çizim işlemlerini bir araya getirmektedir. Ancak tasarımın en önemli unsuru ona yön verecek olan insan faktörüdür. Bugün, ışıklı kalemler, görüntü işlemcileri, fare(mouse), klavye, optik levhalar tamamıyla kullanıcı rahatlığı düşünülerek oluşturulan ürünlerdir. İleride yararlanılabilinecek biçimde çizimler oluşturulup bunların saklanması, çizim işlerinde sürekli kullanılan standart tasarımlarla bir kütüphane oluşturulması, elle çizim ve hesaplaması saatler alabilecek çizim ve boyutlandırma işlemlerinin yapılması, bunlara bir veri tabanı oluşturulması işlemleri CAD yardımı ile basit olarak gerçekleşebilmektedir [11].

AutoCAD genel amaçlı bir çizim programı olarak çok geniş bir yelpaze içerisinde herhangi bir disipline özgü komutlarla kullanıcıyı kısıtlamadığı gibi, açık mimarisi ile istenilen yönde özelleştirilebilmektedir. AutoCAD tüm mimar, mühendis, tasarımcı, grafiker ve kısaca tasarım ve çizim ile ilgili her disiplin tarafından rahatlıkla kullanılabilecek bir programdır. Bugün ülkemizde ve dünyada makine mühendisliğinden güzel sanatlara, mimarlıktan tıbba, şehir planlamadan reklamcılığa, haritacılıktan elektroniğe, uzay araştırmalarından deniz bilim araştırmalarına kadar her alanda AutoCAD'ten temel tasarım ve çizim paketi olarak yararlanılmaktadır [12].

AutoCAD üzerinde referans dosyası kullanımı projelere sağlanan en önemli yardımlardan birisidir. Bir tasarım üzerinde çalışırken, bir başka tasarımı referans olarak çağırıp yeni ya da ilgili tasarımda o çizimden yararlanma olasılığı vardır. Örneğin, bir mutfak dolabı çizimi üzerinde çalışan orman endüstri mühendisi dolaba ait bir parçanın çizimini daha önce yapılmış bir çizimden alıyor ya da kendi üretim resmi olarak hazırlıyor olabilir. Bu durumda parçayı dolap çizimi içerisinde referans alarak ufak değişiklikler yapmak yeterli olacaktır. Dolap resmine girildiğinde parçanın son hali otomatik olarak belirecektir. Böylelikle kavramsal tasarım aşamasında değişiklikler ve yenilemeler kolay şekilde sağlanacaktır [12].

AutoCAD'in yaygın olarak tercih edilmesinin sebebi, gerek 2, gerekse 3 boyutlu tasarım ve çizim için sağladığı olanaklar ve kullanım kolaylığıdır. AutoCAD gerçek bir 2 ve 3 boyutlu tasarım ve çizim yazılımıdır. AutoCAD programında komut ve

menü alanlarında çevrilmiş olan alan kullanıcının tasarım ve çizim alanıdır. Kullanıcı bu pencerede tıpkı 2 boyutta çalıştığı şekilde 3 boyutta da kendi çizim uzayını oluşturup 3 boyutlu tasarım ve çizimler yapabilir. AutoCAD ile çizim yaparken kullanıcı tanımlı çizim elemanlarını kullanabileceği gibi değişik düzenleme komutları ile istediği karmaşıklıkta geometrilerde oluşturabilir. Çizgi, daire, elips, yay gibi basit çizim elemanları 2 ya da 3 boyutlu ‘‘spline’’ eğriler, 3 boyutlu yüzeyler, koni, silindir gibi 3 boyutlu ve karmaşık elemanların kullanımı ile 2 boyutlu karmaşık profilleri döndürerek, kesitleri uzatarak ya da 3 boyutlu eğrilerin arasını dokutarak karmaşık 3 boyutlu yüzeyler elde etmek AutoCAD için oldukça kolaydır [12].

AutoCAD ile çizim düzenleme ve değiştirme komutları ile bir çizimden benzeri çizimleri türetmek ya da saatler sürebilecek bir yenileme işlemi de 1-2 dakika içerisinde gerçekleştirilebilmektedir. AutoCAD tüm çizim elemanlarını kopyalayabilir, taşıyabilir, birleştirebilir. Bir teknik resmin ya da perspektif görünüşün çarpıcılığını çizimin düzeni kadar etkileyen unsurlarda çizgi ve tasarımın kalitesidir. AutoCAD kütüphanesinde yazı stili barındırır [13].

AutoCAD’in sahip olduğu bir diğer önemli özellik otomatik ölçülendirmedir. Doğrusal ve açısal ölçülendirme, çap ve yarıçap ölçülendirme AutoCAD ile çok kolaydır. Ölçü komutuna girip ölçülendirilmesini istediğiniz nesneyi göstermeniz ve ölçü çizgisini konmasını istediğiniz yeri belirtmek yeterlidir [14].

Katı modelle ilgili AutoCAD’ten alınabilecek bilgiler sadece geometrik bilgiler ile sınırlı değil, tasarımcı mühendislerin normal yöntemlerle hesaplamak için saatler, bazen günler bazen de hesaplayamayıp, sadece yakın değerler ile yetinecekleri tüm mekanik bilgileri anında elde etmek ve görüntülemek mümkündür [14].

2 ve 3 boyutlu olarak yaratılan görüntüleri renkli olarak kaplamak da mümkündür. Kaplamalar AutoCAD ile otomatik olarak yapılabilmektedir. Çizimlerle ilgili yazılı veriler daha ileri uygulamalarda kullanılmak üzere AutoCAD ANSI/ISO standardı olarak veri tabanı yönetim sistemleri ile iletişim kurmayı sağlamaktadır. AutoCAD

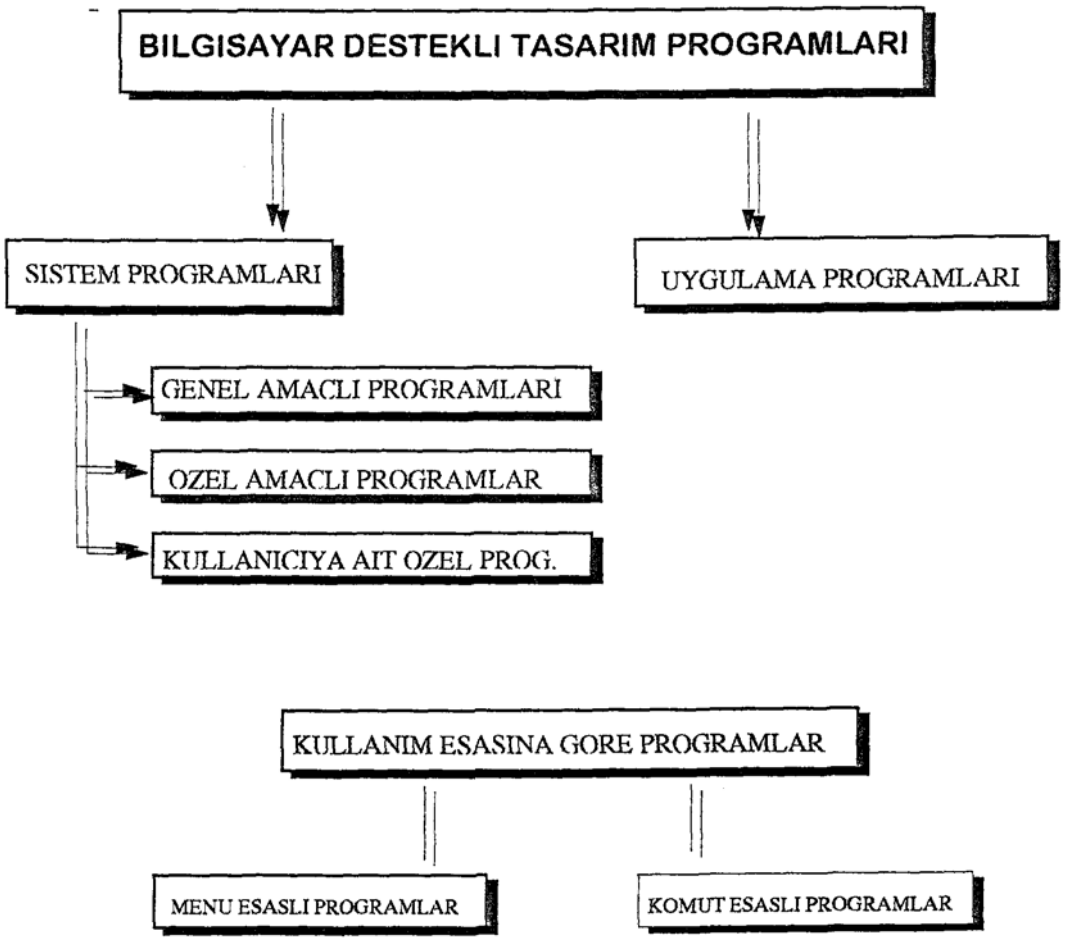
ile çizilen teknik resimlerin bilgileri çeşitli biçimlerde ihraç edilebilir veya veri tabanlarında saklanabilir [13]. AutoCAD sadece veri tabanı yönetim sistemleri ile değil masa üstü yayıncılık sistemleri ile de 2 yönlü kurabilir. Görüntüleri PostScript dosyası olarak okuyabilmesi ve aktarabilmesi AutoCAD'in kullanım alanını daha da genişletmektedir [14].

2.1.1. Bilgisayar destekli tasarım programları

Bilgisayar destekli tasarım için genel olarak iki türlü yazılım gereklidir. Bunlar:

a. Sistem programları

b. Uygulama programları (Şekil 2.1)



Şekil 2.1. Bilgisayar destekli tasarım programları [6]

a. Sistem Programları

Bilgisayar sisteminde bilgisayarın çalışmasını yöneten işletim sistemi (operating sistem) içerirler. İşletim sistemi bilgisayar ile çevre birimleri arasında iletişim sağlayan, belleklere bilgi saklama ve bu bilgileri geri alma, paylaşma, bunların güvenliğini sağlama, bellekler arası bilgi transfer etme ve bellek yönetimi gibi işlemleri yapan bir programdır [6].

Bilgisayar destekli tasarımda en çok kullanılan işletim sistemi MS-DOS (Microsoft Disk Operating System), CP/M (Control Program For Microcomputers), Unix ve Windows'dur.

b. Uygulama Programları

Bilgisayar destekli tasarımda genel amaçlı programlar birçok alandaki isteğe cevap verirler ve bu tip programlar büyük yazılım (software) firmaları tarafından üretilirler ve çok çeşitli alanlara yayılmışlardır. Bu tür programlar da menü bazında çalışanlar menülerden seçim yaparak işlem gerçekleştirilir. Komut bazında çalışan CAD programları da vardır. Bu tip programlar mühendislik, mimarlık, mobilya ve dekorasyon alanlarına yöneliktir. Bunlar iki boyutlu, üç boyutlu tasarım yaparlar. İki boyutlu CAD programları, iki boyutlu ve bazıları ile de tel kafes şeklinde üç boyutlu çizim yapılabilir. Üç boyutlu olanlar (3D) yüzey, model ve üç boyutlu katı modelleme yaparlar [6].

Özel amaçlı programlar ise kullanımı kolay ve sadece özel bir alana uygulanabilen programlardır. Bu programlar tek bir mutfak tasarımı, banyo tasarımına vb. özel amaçlı hazırlanmış programlardır.

2.2. Bilgisayar Destekli Tasarımda Autolisp'in Yeri

Bilgisayar destekli tasarım (Computer Aided Design, CAD) herhangi bir ürünün tasarımı sırasında tasarımın oluşturulması, geliştirilmesi, analizi ve optimizasyonu

aşamalarında bilgisayarın kullanılması şeklinde tanımlanabilir. Tasarım çalışmalarında bilgisayarın kullanılması, tasarımcının bilgisayarın yeteneklerinden faydalanarak daha kısa sürede, daha iyi tasarımlar yapabilmesini sağlar [6].

Bilgisayar destekli tasarım sayesinde, tasarımcılar (mobilyacılar) yaptıkları tasarımın kontrolünü ve yaratacağı sonuçları bilgisayardan kısa sürede öğrenerek gerekli değişiklikleri hemen yapabilirler. Bu şekilde adım adım ilerlenerek en uygun tasarım elde edilir. Kontrol hesaplarının ve değişikliklerinin bilgisayar sayesinde kısa sürede yapılabilmesi, tasarım aşamasını daha verimli hale getirir [6].

Bilgisayar destekli tasarım sistemleri, gelişmiş grafik özelliklere sahiptir. Tasarımcı çizmek istediği şekle geometrik bilgileri giriş üniteleri (tuş takımı, mouse, vb.) kullanarak bilgisayara verebilir. Bilgisayar hafızasında daha önce yüklenmiş olan bilgileri kullanarak, kullanıcının istediği şekli ekranda oluşturur. Grafik sistemin özelliklerine bağlı olarak bazı bilgisayar sistemlerinde üç boyutlu çizimler oluşturulabilir. Yapılan çizimler kullanıcının verdiği bilgilere uygun olarak bilgisayar tarafından boyutlandırılabilir ve bu sayede istenilen kısımlardan gerekli kesit ve detaylar alınabilir [6].

Birden fazla parçadan meydana gelen montaj resimleri elde edilebilir. Bu montaj resimleri için malzeme ve parça listesi alınabilir. Parçalar üzerinde zorlanmaları ve bunlara bağlı şekil değiştirmelerini gösteren diyagramlar elde edilebilir. Ekranda oluşturulan her türlü çizimler ile tasarıma ait diğer bilgiler istendiği takdirde yazıcı ve çizici gibi çıkış üniteleri kullanarak kağıda aktarılabilir [6].

Bilgisayarın, yapılan çalışmalarda elde edilen sonuçları magnetik ortamda saklayarak istenildiğinde yeniden kullanıcıya sunma yeteneği sayesinde daha önce yapılan tasarımlar gerektiği zaman değişen şartlara uygun olarak kolayca yenilenebilir [6].

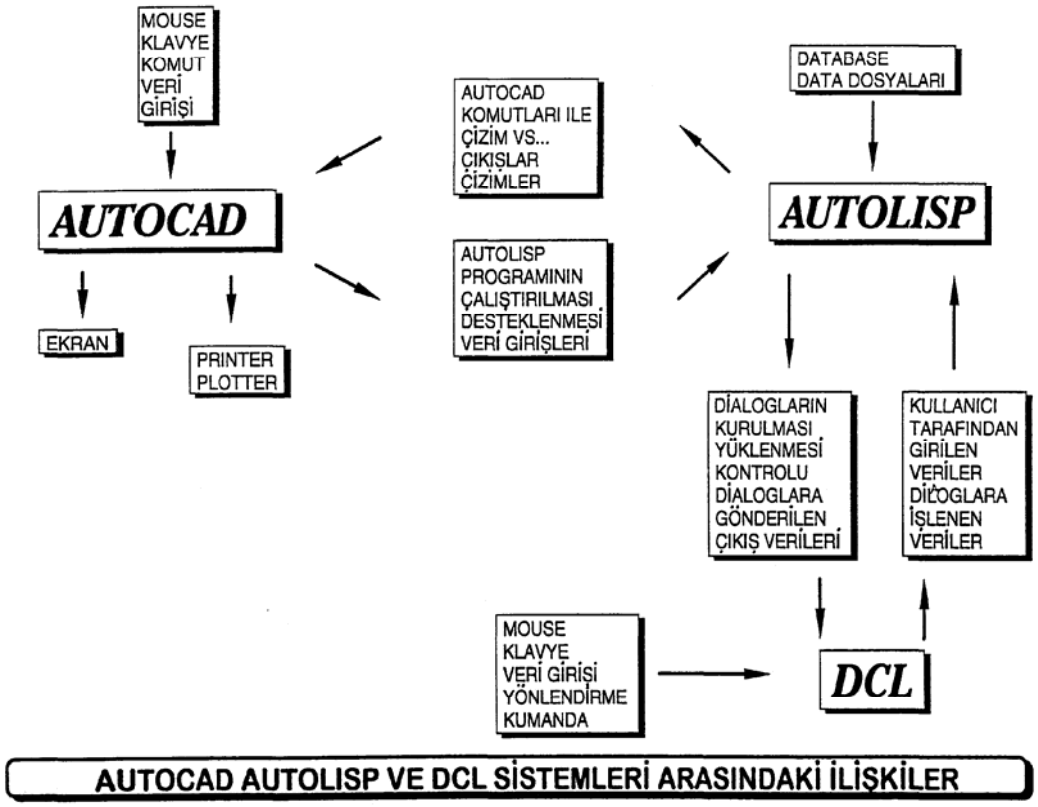
2.3. Autolisp Programlama Dili

AUTOLISP, LISP programlama dilinin AutoCAD içine yerleştirilmiş bir uyarlamasıdır. AUTOLISP yapay zeka çalışmalarında kullanılan bir programlama dilidir. AUTOLISP ile AutoCAD'e yeni komutlar eklenmesi, kişiselleştirilmesi ve artan bir verim elde edilmesi mümkündür. Yeni komutlar eklenmesi demek; AUTOLISP fonksiyonlarını kullanarak hazırlanan program dosyaları AutoCAD ortamında çağrılarak kullanılmasıdır [7].

AUTOLISP programını yazmak için bir tane işlemci (Text editör) programı kullanılır. Bunun için AutoCAD'den çıkmadan kullanılabilen ve DOS içinde bulunan EDLIN veya bağımsız olarak çalışan Word Star, Word gibi programlar kullanılabilir. Ancak yazılan programlar AutoCAD tarafından anlaşılabilmesi için ASCII formatında bir dosyada saklanması gerekir. Saklanan dosya *LSP uzantılı olmalıdır.

AutoLISP ve AutoCAD ile iyi derecede grafik uygulama çalışmalarının yapılması yanında, yüksek seviyeli dillerdeki gibi makro düzeyde program yazımı mümkündür [7].

AutoCAD, AUTOLISP ve DCL Sistemleri Arasındaki ilişkiler Şekil 2.2'de verilmiştir.



Şekil 2.2. AutoCAD, AUTOLİSP ve DCL Sistemleri Arasındaki İlişkiler [6]

2.3.1. Kuruluş mantığı

AutoLISP programlarını yazıp çalıştırmadan önce DOS ve WINDOWS işletim sistemi kullanıcılarının, AutoLISP için gerekli bellek ayırma işlemini yapmaları gerekir. DOS'un "SET" komutu kullanılarak bellek ayarlaması yapılır. AutoLISP için ayrılan bellekte program yazmak için Lispheap, yazılan programı çalıştırmak için lispstack ve daha fazla çalışma boşluğu bırakmak için Acadfreeram alanları bırakılır. Lispheap ve Lispstack alanları için bırakılan boşluk 45000 byte'ı geçmemelidir. Acadfreeram için ise ayrılacak olan alan 24 byte'ı geçmemelidir. AutoLISP programı çalıştırılmadan önce set komutları kurulmalıdır. Bu komutların kurulmasında üç yöntem vardır [7]. Bunlar :

1. Her zaman AutoCAD çalışmasında AutoLISP'i hazır kullanım için set komutları AUTOEXEC.BAT dosyası içerisine aşağıdaki gibi yazılır.

   set lispheap = 39000

   set lispstack = 5000

   set acadfreeram = 24

   set acad = \ lisp

2. İkinci bir yazılış, ACAD direktöründe özel bir batch dosyası içerisinde olur.

   cd \ acad

   set lispheap = 39000

   set lispstack = 5000

   set acadfreeram = 24

   set acad = \ lisp

   acad

3. AutoCAD çalışmasında her defasında set komutları yazılır ve bilgisayar kapatılınca özelliği biter. Yeni bir AutoCAD çalışması için bu satırları yeniden yazılmak zorundadır [7].

   set lispheap = 39000 <ENTER>

   set lispstack = 5000 <ENTER>

   set acadfreeram = 24 <ENTER>

   set acad = \ lisp <ENTER>

2.3.2. Program yazma editörü

AutoLISP programlan herhangi bir yazım (text) editöründe yazılırlar. En çok kullanılan, DOS altında bulunan EDLIN yazım editörüdür. Bunun avantajı, DOS

komutu ve uzantısı ".COM" olduğundan AutoCAD ortamından geçişi kolaydır. AutoCAD çalışıyor durumda iken EDLIN yazım editörüne geçiş yapılır ve AutoLISP programı yazılıp "save" yapıldıktan sonra AutoCAD ortamı yeniden güncelleşir. AutoLISP programı yazılmadan önce EDLIN.COM dosyası AutoCAD'in bulunduğu klasörün içerisine kopyalanmalıdır. AutoCAD kalasörü genellikle ACAD altında bulunur. Dolayısıyla EDLIN.COM dosyası C:\ACAD> klasörü altına kopyalanır. ACAD.PGP adındaki diğer bir yazım editörü, EDLIN'in yerine başka bir yazım editörünün de kullanılabilirliğini sağlar. Basit bir AutoLISP programı, AutoCAD güncel durumda iken aşağıdaki gibi yazılır [7].

Command : EDIT <ENTER>

veya

Command : SH <ENTER>

Dos file Command : Edin Denemel.LSP <ENTER>

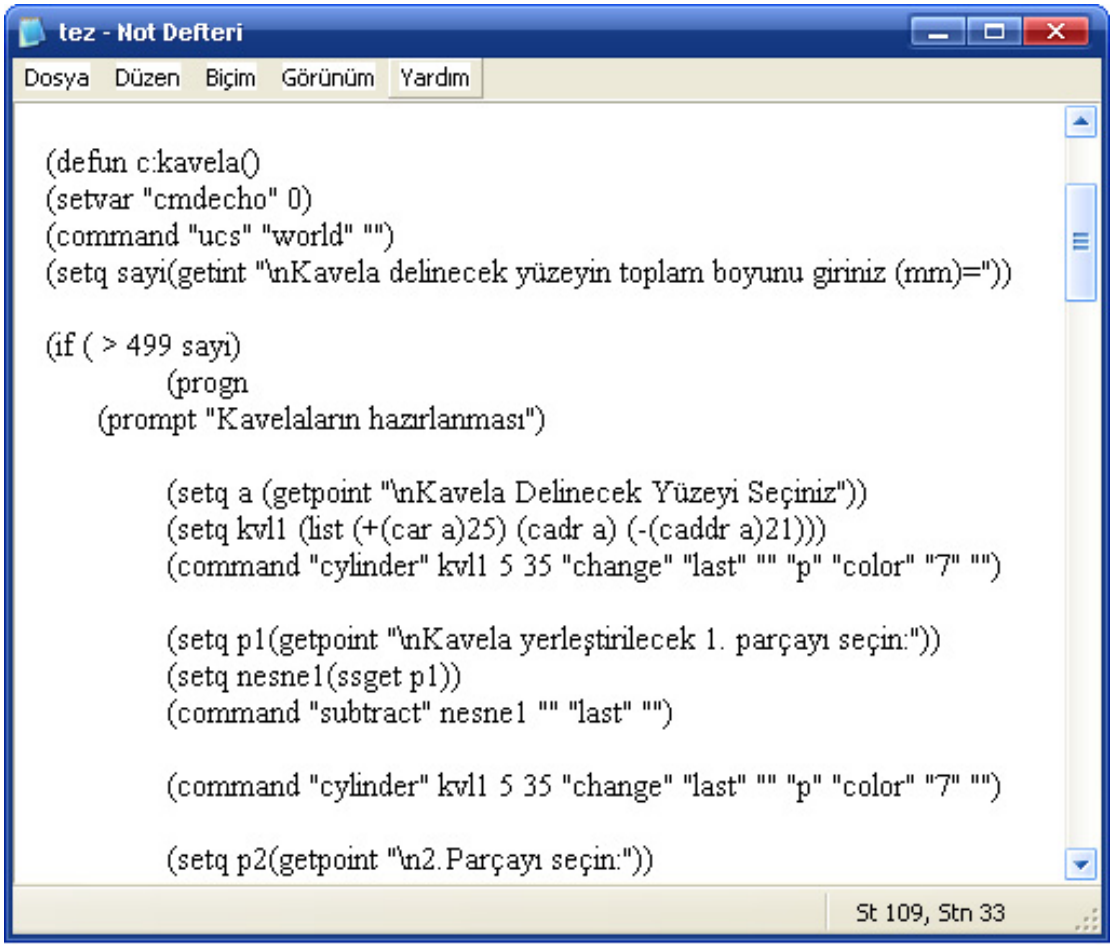
Dos altındaki EDLIN, SH, SHELL yazım editörlerinden hangisi kuruldu ise o text dosyası güncellesin

fdefun C: denernel ( ) <ENTER>

(prompt "Benim programım") <ENTER>

) <ENTER>

Yukarıdaki üç satır yazıldıktan sonra yazım editöründen "SAVE "edilip "EXIT" ile çıkılarak AutoCAD çizim ortamına dönülür. Ayrıca kullanıcı WINDOWS ortamında çalışıyorsa "NOTEPAD" editörünü de kullanabilir. Notepad editörü ve AutoCAD programı aynı anda kullanım özelliğine sahiptirler. AutoCAD ve notepad'in WINDOWS pencerelerinde aynı anda her ikisinin de görünür olarak çalışması programlama hızının ve esnekliğinin artmasını sağlayacaktır. Text editörünün yüklenmesini veya kapatılmasını beklemeksizin aktif hale gelmesini istediğiniz pencerenin herhangi bir yerine tıklanarak hemen uygulama başlatılabilir (Şekil 2.3) [7].



Şekil 2.3. Lisp dosyasının notepad'te açılması.

2.3.3. Çalışılan programı açıklayıcı satırlar

AutoLISP programlama dilinde açıklayıcı bilgileri yazmak için noktalı virgül (;) kullanılır. Program yüklendiği zaman hangi aşamada nelerin bulunduğu, açıklayıcı satırlardan anlaşılır. Örneğin bir dişli çark çizimi programında girdiler, hesaplamalar, açı hesaplamaları, çizim hesaplamaları ve dişli çizimi aşağıdaki gibi açıklayıcı satırlarda baslık gibi yazılır [7].

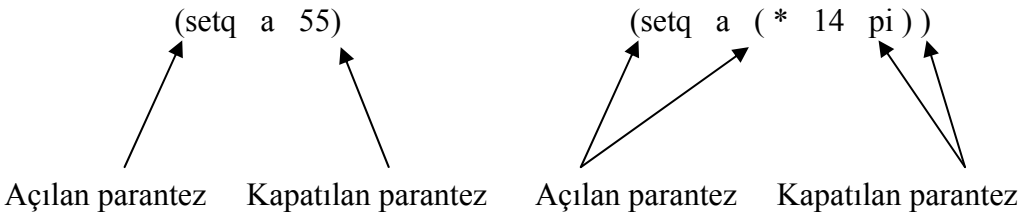
(defun KAVELA ( )

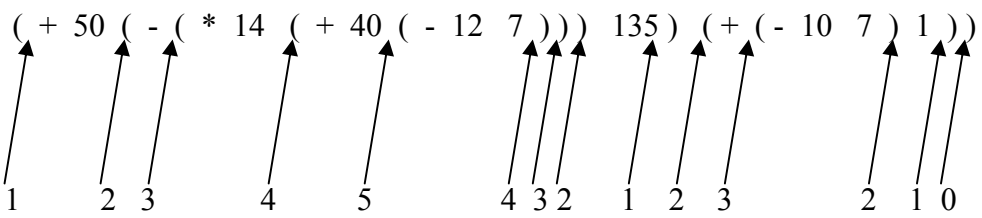
; Kavela çizimi için parametre girdileri
……
;;; Kavela boyutlarının hesaplanması
……
;; Çizim noktası açılarının hesabı

```
……
;;;;;;  Çizim  noktaları  hesabı
……
;;; AutoCAD komutu üne ile dişli çizimi
……
```

2.3.4. Parantezlerin kullanımı

AutoLISP programında parantez en temel ve en önemli karakterlerden birisidir. Tüm komutlar parentez içerisinde kullanılır. Açılan parantez kadar kapatılmış parantez olmalıdır. AutoLISP programı parantezle başlar ve parantezle biter. Parantez hatalarının çabuk kontrol edilebilmesi, açılan ve kapatılan parantezlerin sayılmasıyla yapılabilir. Açılan her parantez için bir sayı eklenirken kapatılan her parantez için bir sayı eksiltilir. Son kapatılan parantez sayısı "0" ise açılan parantez kadar parantez kapatılmış demektir. Aşağıdaki örnekler kontrol mekanizmasının nasıl çalıştığını göstermektedir [7].





Örnek:

```
(defun C:dişli ( )
        ;;; dişli çizimi girdileri
        ( setq nl  ( getpoint  "\n  bir nokta gir ya da seç" )
        n2 ( getreal "\n bir nokta x , y gir" )
        )
)
```

Örnek:

```
(defun  C:disli  ( )
        ;;;    dişli   hesapları
        ( setq m   3 )              ;   modül   3
        ( setq   z   zl )           ;   diş   sayısı
        ( setq  do   ( *   m   z )
                da   ( +   do    ( *   m   z ) )
          )
)
```

AutoLISP ile program yazarken yukarıdaki örneklerde olduğu gibi her açılan parantezin kapatılması, aynı satırda veya bir altındaki satır ve aynı hizada olursa program daha iyi anlaşılır olur. Eğer açık bir paranteze ya da kapatılmış bir paranteze rağmen program çalıştırılırsa aşağıdaki mesaj gelecektir [7].

```
Command  :  (  load  "DISLI"  )  <enter>
1>  (açık  parantez  sayısı  2  ise  mesaj  2>  şeklinde  olur)  ) <enter>
```

Burada kullanıcıya açık parantez sayısı gösterilmektedir. Program parçasında gerekli düzenleme yapıldıktan sonra program çalışır hale gelecektir. Kapatılmamış parantez sayısı birden fazla ise aynı mesaj tekrar eder [7].

2.3.5. Tırnak işareti

Tırnak işaretleri yazı olarak okunması İstenen verilerin işaretlenmesi için kullanılırlar. Tırnak işaretleri sembol veya fonksiyon değildirler. İlk tırnak işareti açılmış olan tırnak işareti olarak düşünülür. Girilen bütün veriler tırnak işareti kapatılıncaya kadar yazı modunda ekranda görünecektir. Tırnak işaretleri gözden kaçırıldığı veya unutulduğu zamanlarda bir fonksiyonmuş gibi algılanacaklarından programda yanlışlıklar meydana getireceklerdir. Tırnak işaretleri aynı zamanda < enter > anlamında da kullanılırlar. Bu kullanım şekli tezin ileriki bölümlerinde anlatılacaktır [8].

Command : (setq a “ok”) <enter>

“ok”

Command : (setq a “ok”) <enter>

1>”

1> <enter>

“ok) \ n”

2.3.6. Veri tipleri

AutoLISP ile programlamada kullanılabilecek veri tipleri aşağıda çıkartılmıştır.

- Tamsayılar
- Reel sayılar
- Yazı dizileri (strings)
- Listeler
- AutoCAD Değişkenleri
- AutoCAD Objeleri (entities)

- AutoCAD ortamında seçilmiş objeler.

Tamsayılar, -32768 ile +32767 arasında olabilir.

1 20 -421 45 31702 -650 890 gibi

Reel sayılar,

7,982 – 1,015 3,14 543,68 -765,5 0,45

şeklinde gösterilen sayılara reel sayı denilmektedir. Reel sayıları fonksiyonlar içinde kullanılırken l den küçük değerlere sahiplerse ondalık noktanın soluna 0'ın yazılması program akışı içinde meydana gelebilecek karışıklıkları önleyecektir. Örneğin; .25 yerine 0,25 - 1 yerine 0.1 gibi.

Yazı dizileri, tırnak içinde yazılması gereken karakter dizileridir, uzunluk sınırlaması yoktur.

" BİRİNCİ NOKTA"

"uzaklık"

"Ad Soyad " gibi.

Listeler, değişkenlerden oluşan gruplardır. ( ) içinde yazılırlar değişik amaçlar için kullanılabilirler. En fazla kullanım alanı nokta koordinatlarının tanımlanmasındadır.

(X Y) (15 23) (53.1)

(X Y Z) (2 4 1) (5.25 0.20 2.2) gibi

AutoLISP'de çok kullanılan iki fonksiyon vardır. Bunlardan biri "pi" fonksiyonu, diğeri de nil fonksiyonudur. "Pi" fonksiyonunun değeri önceden belirlenmiştir. Bu

değer 3,1415926'ya eşittir. Yazılan programlarda bu değeri kullanmak istediğinizde "pi" yazılması yeterli olacaktır.

Nil fonksiyonu "hiç" değerine eşittir. Program içinde setq fonksiyonu kullanılarak değer atanmamış tüm değişkenlerin değeri nil'dir. Setq fonksiyonu kullanılarak değer atanmış bir değişkeni programın daha sonraki satırlarında yine setq fonksiyonu kullanılarak nil değeri atanırsa değişkenin önceki değeri boşaltılmış ve değişkene nil değeri atanmış olur. Prompt gibi fonksiyonlar da görevlerini yaptıktan sonra nil değerini verirler [8].

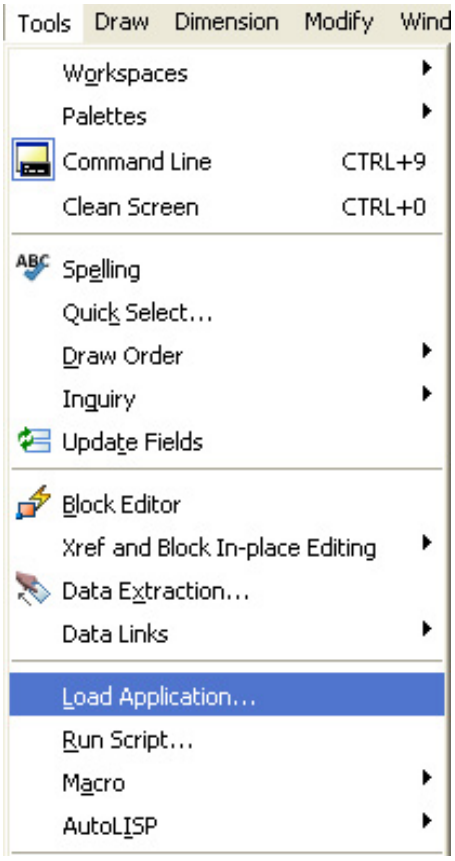
2.3.7. Programların yüklenmesi

AutoLISP programlama dilinde yazılan programın AutoCAD çizim ortamında çalıştırılması için öncelikle yüklenmesi gerekir. Yükleme işlemi İki değişik şekilde yapılabilir. Birincisi AutoCAD ortamında Command satırına aşağıdaki şekilde program adı yazılır. Program ya da dosya adı DISLI ve çalıştırılacak komut da DISLI ise,

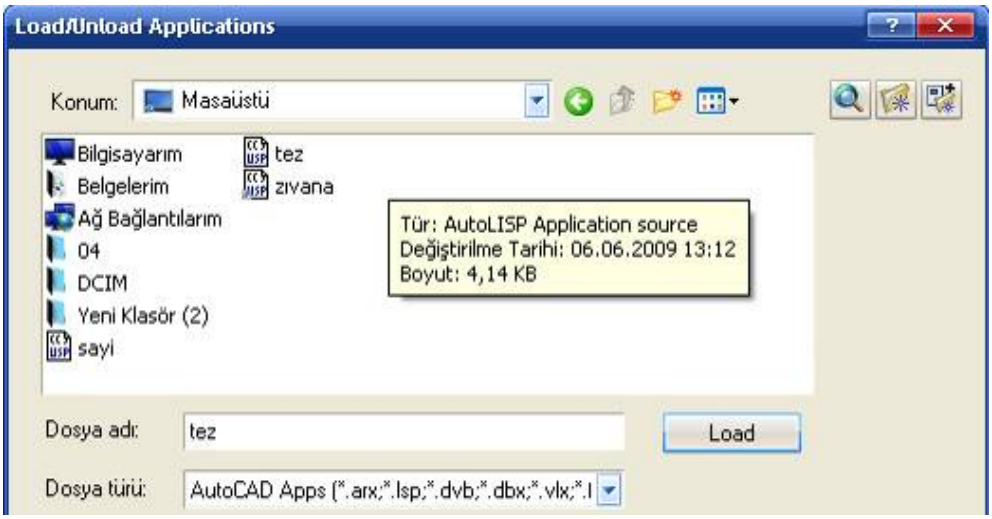
Command : (Load "DISLI") <enter>

Şeklinde program yüklenir. Program da açık kalan parantez ve hata yoksa, "Command" satırı boş olarak tekrar ekrana gelir. Bu aşamadan sonra Command satırına "Command : DISLI" yazılarak "<enter>" tuşuna basıldığında DISLI programı çalıştırılır. AutoLISP programında dosya yüklenmesi yukarıda olduğu gibi parantez içerisinde yapılmaktadır [8].

AutoLISP programının ikinci yükleme şekli ise "file" menüsü altındaki "Application" seçeneğinden gerçekleştirilir. Bu AutoCAD 12 programı için geçerli yükleme şeklidir. AutoCAD 14 sürümünde ise bu seçeneklere "Tools" menüsü altındaki "Load Application" komutuyla ulaşılır. Seçimler yapıldıktan sonra ekrana .LSP uzantılı dosya adlarını veren bir diyalog kutusu gelir. Bu diyalog kutusundan dosya adları ve istenen dosya işaretlenip "load" düğmesi seçilerek AutoLISP programları (bir ya da birden fazla) çizim ortamında yüklenmiş olur. Daha sonra Command satırına program adı yazılıp enter tuşuna basılarak program çalıştırılabilir.



Şekil 2.4. Tool menüsü load application.



Şekil 2.5. Load application.

2.3.8. Autolisp'in sürekliliği

AutoCAD 14 sürümünden önceki sürümlerde bütün AutoLISP değişkenleri ve sembolleri sadece açılan bir dosya içerisinde kullanılabiliyordu. Bu, bütün değişkenlerin ve sembollerin görevleri o dosyadan ayrıldığında bitiyor anlamını taşıyordu. Yani semboller ve değişkenler sadece o anda geçerli olan dosya içerisinde kullanılabiliyordu. Yeni bir dosya açıldığında eğer eski AutoLISP dosyaları kullanılacaksa, tekrar yüklenmeleri gerekiyordu [9].

AutoCAD 14 sürümünün kullanılmaya başlanılmasıyla AutoLISP değişken ve sembollerinin görevlerinin AutoCAD programından çıkılıncaya kadar sürmeleri sağlanmıştır. Bu, AutoLISP sembol ve değişkenlerinin bir çizim dosyasından diğer bir çizim dosyasına geçildiğinde geçilen çizim dosyasında da geçerli olması anlamını taşır. Bu geçerlilik AutoCAD ortamından çıkılıncaya kadar korunur. Bu anlatılanların gerçekleştirilebilmeleri için "Tools" menusunun altında "Preferences" diyalog kutusundan "Compatibility" düğmesine tıklanarak buna ait diyalog kutusunun en alt kısmında bulunan "Reload AutoLISP between drawings" kutucuğu işaretlenmeli veya iptal edilmelidir [9].

2.3.9. Defun foksiyonu

Defun, ana programı veya ana programa bağlı alt programları yazmak için kullanılan en temel fonksiyondur. Defun'un (Define Function) kelime anlamı, fonksiyon tanımladır. Bir sol parantez ve ardından defun terimi yazıldıktan sonra tanımlanacak fonksiyonun veya programın adı yazılır. Defun terimi ile fonksiyon adının arasına C: yazıldığında tanımlanan fonksiyonu bir AutoCAD komutu gibi kullanmamız sağlanmış olur. Defun fonksiyonunun sözdizimi;

(defun [c:] fonksiyon adı ([argümanlar / değişkenler]) Program atamaları ve değişkenlerin tanımlanmaları)

şeklindedir. Yukarıdaki örnek yazılımda, bir fonksiyon tanımlanırken yazılması zorunlu olmayan elemanlar ( ) içinde yazılmışlardır. Noktalar halinde gösterilen satırlarda, bu fonksiyon içinde kullanılacak gerek standart gerekse yine kullanıcı

tarafından tanımlanan alt fonksiyonlar bulunabilir. Defun teriminden sonra C: kullanılmadan tanımlanacak bir fonksiyon, bir alt fonksiyon olarak kabul edilir. Fonksiyon adından sonra argümanların veya değişkenlerin yazılması şart değildir [5].

Örneğin fonksiyon adı DISLI ve alt fonksiyonu da DDUZ olan bir fonksiyonun tanımlaması aşağıdaki gibi yapılır.

```
( defun C: DISLI ( )
    ;;; bu kısımda dişli boyutları hesaplanır.
    ……
( DDUZ ) ;;;DDÜZ programını, güncelleştirir.
)
    ;;; düz dişli hesaplarına yapan program
    ……
( defun DDUZ (  )
```

2.3.10. Command foksiyonu

Bu fonksiyon AutoLISP'in en temel fonksiyonlarından birisidir. Bu fonksiyonun görevi AutoLISP ile AutoCAD arasındaki ilişkiyi sağlamaktır. Command fonksiyonu, hazırlanmış, olan LISP programlan içerisinde AutoCAD komutlarının çalıştırılmasında kullanılır. Command fonksiyonunun sözdizimi;

```
( Command   [ değişkenler ]    . . . )
```

şeklindedir. Burada kullanılan değişkenler " " içinde yazılması gereken AutoCAD komutları, bu komutlara ait alt komutlar ve iletilere verilmesi gereken cevaplardan oluşur.

Örneğin;

```
(Command   "line"  Al   A2   " " )
```

gibi bir fonksiyon tanımlanmış olsun. Bu fonksiyon ile AutoCAD'in LINE komutu çağırılarak Al ile A2 noktaları arasına bir çizgi çizilmesi sağlanır. Al ve A2 noktalarının program içerisinde daha önce tanımlanmış olması gerekir. Boş bir girdi (" ") klavyeden girilen boş değere eşittir ve Ctrl+C'nin girilmesiyle aynı anlamı taşır. Bu işlemi kullanıcı AutoCAD komutlarından çıkmak veya komutları yarıda kesmek için kullanır. LINE komutu Birbiri ardına çalışan bir komuttur. "From point" girildikten sonra kullanıcının ardışık noktalar gireceğini varsayarak "to point" İletisini sürekli tekrarlar. LINE komutunu tamamlamak için "to point" İletisine boş bir yanıt vermek veya Ctrl + C ile komutun çalışmasını kırmak gerekir. Yukarıdaki örnekte de Al ve A2 noktalarından sonra LINE komutunun bitirilmesi için (" ") ile boş girdi yapılması sağlanmıştır. Çalışma prensibi LINE komutu gibi olan tüm komutlarda komutu bitirmek için " " ile boş girdi yapılması gerekir [10].

Command fonksiyonu ile AutoCAD komutlarının çalıştırılması sırasında dikkat edilmesi gereken en önemli nokta, Command fonksiyonu sonrasında AutoCAD komutunun tırnak içinde yazılması ve sonra da değişkenlerin doğru sırada komutun arkasından girilmesidir. Yani komut AutoCAD ortamında normal olarak çalıştırıldığında hangi soruları soruyorsa, Command fonksiyonu kullanılırken bu sıralama dikkate alınmalı ve değişkenler AutoCAD komut iletilerine, sorularına cevap olabilecek sırada yazılmalıdır [10].

Örneğin "LINE" komutu "from point" iletisinden sonra ardışık olarak "to point" iletisini görüntüler. "Arc" komutu çalıştırıldığında default olarak 3P yöntemini kabul eder ve başka yöntem belirtilmezse ardışık olarak üç noktanın girilmesini ister ve bu üç noktayı kullanıp bir yay çizerek, komuttan çıkar. Örneğin arc komutunu bitirmek için çift tırnak işaretine (" ") gerek yoktur, çünkü üç noktanın da gösterilmesinin ardından komut otomatik olarak bitirilir [10].

```
( Command "arc" Al A2 A3 )
```

yazıldığında AutoCAD komutlarından "arc" komutu çalıştırılarak tanımlanan noktalardan geçen bir yay çizecek ve komutu otomatik olarak bitirecektir. Bu komutu bitirmek için boş girdiye (" ") gerek yoktur [10].

Örnek:

```
Command: (Command “line”  “2,2”  “4,4”  “” )              <ENTER>
LINE from point      : 2 , 2
To point             : 4 , 4
To point             :
Command              : nil
```

Yukarıdaki örnekte Command fonksiyonu ile AutoCAD ortamından "line" komutu çağırılarak, line komutu kullanıcıdan hangi değerlerin girilmesini istiyorsa o değer ve değişkenler line komutunun arkasından tırnak içerisinde yazılır. Yani LINE from point yerine "2,2", To point yerine de "4,4" girilmiştir. "" işareti, line komutundan çıkmak için kullanılır ve AutoCAD ortamında Command satırını yeni komut girilebilecek şekilde boşaltır [10].

Örnek:

```
Command : ( setq nokta1 '( 4 2 ) )                                  <ENTER>
Command : (Command “line”  ' ( 6  4 )  noktal ' ( 4 6 )  “c” )      <ENTER>
LINE from point      : 6,4
To pomt              : 4,2
To point             : 4,6
To point             : c
Command              : nil
```

Öncelikle Command fonksiyonuyla line komutu çağırılır. “LINE from point” yerine '(6 4) şeklinde verilmiş listeden X değeri 6, Y değeri 4 olan nokta alınır. "To point"

yerine de nokta 1 alınır (bu nokta bir önceki satırda '(4 2) olarak tanımlanmıştı). Yine "To point" yerine '(4 6) listesi atanır. Burada kullanılan "c" ifadesi, çizilen en son çizginin son noktasının en başta çizilen çizginin ilk noktasıyla birleşmesini sağlamaktadır. Bu işlemden sonra Command fonksiyonu nil'e dönüşecektir [10].

Örnek

(Command “circle” ' (1 1 0) 2 . 5)

Merkez noktası (1 1 0) ve (yani X değeri 1, Y değeri 1 ve Z değeri 0) yarıçapı 2.5 olan daire çizer.

(Command “_circle” ' (2 2) “_d” 4)

Circle komutuna girilerek merkezi 2,2 ve çapı 4 olan daire çizdirilir.

Birden fazla AutoCAD komutunun kullanılması gerektiğinde her komut için Command fonksiyonunun tekrarlanmasına gerek yoktur. Önemli olan bir sol parantez ile başlatılmış olan Command fonksiyonunun içinde komutlar kullanıldıktan sonra sağ parantez ile fonksiyonun kapatılmasıdır [10].

Örnek:

(Command “ line ” Al A2 “ ”

“ arc ” Al noktal nokta2

“ line ” nokta3 Al “ ”

Bu örnekte ilk önce line komutu çalıştırılmaktadır. Al ile A2 noktaları arasına bir çizgi çizildikten sonra " " ile line komutundan çıkılmaktadır. "Arc" ile arc komutunun çalışması sağlanmakta, Al noktasında başlayan, noktal den geçen ve nokta2 de biten bir yay çizilmektedir. "Line" ile line komutu tekrar çalıştırılmakta ve nokta3 ile Al araşma bir çizgi çizildikten sonra boş girdi " " ile line komutundan çıkılmaktadır [11].

Örnek:

Aşağıdaki örnek programda kullanıcıdan dört girdi istenmektedir. 1.(nokta 1) ve 3.(nokta2) girdiler çizilecek çemberlerin merkez noktalarının yerini, 2.(yarıçap 1) ve 4.(yarıcap2) girdiler bu çemberin yarıçap değerlerini ister. Daha sonra çemberlerin merkez noktalarını birleştiren bir doğru çizilerek programı bitirir [11].

(defun c:cember (/ nok1 ycap1 nok2 ycap2)

(setq noktal (getpoint "\nBIRINCI ÇEMBERİN MERKEZİ:")) (setq yarıcap1 (getreal "\nBIRINCI CEMBERIN YARICAPI :")) (setq nokta2 (getpoint "\nIKINCI ÇEMBERİN MERKEZİ :")) fsetq yarıcap2 (getreal "\IKINICI CEMBERINYARICAPI :")) (Command "circle" noktal yarıcapl

"circle" nokta2 yarıcap2 "üne" nokta1 nokta2 ""

Programı yazıp çalıştırdığınızda özellikle "circle" komutu çalışırken ekranın komut alanında circle komutunun iletilerinin akıp geçtiğini görebilirsiniz [11].

Burada tezin ileriki bölümlerinde de mevcut olan bazı komutlardan ve fonksiyonlardan bahsetmek yerinde olacaktır. AutoCAD sistem değişkenlerinden CMDECHO'nun değeri 1 ise komut iletileri ekrana yansıtılır. Yukarıdaki örnekte kullandığımız "CİRCLE" komutunun bütün iletileri komut çalışırken ekranda görüntülenecektir. Bunun istenmediği durumlarda program çalıştırmadan Önce CMDECHO sistem değişkeninin değeri 0'a eşitlenmelidir. CMDECHO, bir sistem değişkeni olduğundan sistem değişkenlerinin değerlerini değiştirmemizi sağlayan SETVAR fonksiyonu ile CMDECHO'nun değeri 0 veya 1 haline dönüştürülebilir. Bu fonksiyonun programa eklenmiş hali aşağıda verilmiştir [11].

(defun c:ceınber (/ nok1 ycap1 nok2 ycap2) (setvar "cmdecho" 0)

(setq nok1 (getpoint "\NBIRINCI ÇEMBERİN MERKEZİ :">)

(setq ycap1 (getreal "\nyarıcapı :"))

(setq nok2 (getpoint "\nIKINCI CENBERIN MERKEZİ :"}),

(setq ycap2 (getreal "\nyarıcap :")}

(Command "circle" nok1 ycap1

"circle" nok2 ycap2

"üne" nok1 nok2 ""))

Yukarıdaki program çalıştırıldığında AutoCAD sistem değişkenlerinden biri olan CMDECHO'nun değeri 0 'a eşitleyecek ve program, diğer fonksiyonları yerine getirmeye başlayacaktır. Bu fonksiyonun değerinin tekrar l'e eşitlenmesi istenirse program içerisindeki Command satırının altındaki satıra yani programın son satırına SETVAR fonksiyonu kullanılarak CMDECHO değişkeninin değeri değiştirilir. Programın son satırına aşağıdaki ifade yazılır [11].

(setvar "cmdecho" 1)

Not: Sistem değişkenleriyle ilgili olarak tezin ekler kısmına bakılabilir.

Örnek:

(defun C: DÖRTGEN ( )

; ; ; bu program dikdörtgen noktalarını tanımlar ve çizer

(setq n1 (getpoint "\n birinci noktanın yerini seç") n2 (getpoint "\n ikinci noktanın yerini seç") n3 (getpoint "\n üçüncü noktanın yerini seç") n4 (getpoint "\n dördüncü noktanın yerini seç")

)

(Command "üne" nl n2 n3 n4 nı “ ”)

)

Dörtgen çizimindeki son satırda nl n2 n3 n4 nı deki son n1 yerine "c" komutu "close" şeklinde kullanılırsa son satırın aşağıdaki gibi yazılması gerekirdi.

(Command "üne" nl n2 n3 n4 "close")

veya;

(Command "üne" nl n2 n3 n4 "c")

3- AutoCAD çalışmasında her defasında set komutları yazılır ve bilgisayar kapatılınca özelliği biter. Yeni bir AutoCAD çalışması İçin bu satırları yeniden yazılmak zorundadır [11].

set lispheap = 39000 <ENTER>

set lispstack = 5000 <ENTER>

set acadfreeram = 24 <ENTER>

set acad = \ lisp <ENTER>

2.3.11. Setq fonksiyonu

Bu fonksiyon AUTOLISP'te değişkenlere bir değer atamak için kullanılan bir fonksiyondur. Bu fonksiyonu kullanma formatı şöyledir.
(Setq <değişken adı- değeri>…) şeklinde yapılır [8].

Örnek: (setq x 4.5) yazıldığında sembol x'i 4.5'e ayarlar. X her yazıldığında gerçek sayı 4.5 değerlendirilir. Setq tarafından bir sembole doğrudan atanan dizgiler için yüz karakterlik bir maksimum uzunluk vardır. Ancak, bir çok dizginin birbirini takip sonucunu bir değişkene atamak için ‘‘strcat’’ fonksiyonunu kullanarak daha uzun dizgiler oluşturulabilir.

2.3.12.Getpoint fonksiyonu

Bu fonksiyon AUTOLISP'in önemli nokta tanımlama fonksiyonlarından biridir. Bu işlev kullanıcının bir nokta girmesini bekler. Bu nokta, mouse yardımı ile ekranda

herhangi bir yere işaretlenerek yapılabildiği gibi, güncel birimler formatında bir koordinattır. Klavyeden girilerek de yapılabilir. Ekrandan veya klavyeden girilen bu noktanın koordinatı ‘‘setq’’ fonksiyonu ile bir değişkene atanarak programda kullanılabilir. Bu fonksiyonu kullanma formatı şöyledir (GETPOINT (NOKTA) (İLETİ)) [8].

Örnek: (Setq P1(GETPOINT’’\n ÇEMBER SOL ORTA NOKTASI : ’’)) ;

Girilen noktanın değerini (koordinatlarınısetq fonksiyonu ile P1 değişkenine atar.

2.3.13. Getreal fonksiyonu

Bu fonksiyon, kullanıcının bir gerçek sayı girmesi için duraklar ve bu gerçek sayıyı işleme koyar. Girilecek gerçek sayı, programın ilerleyen aşamalarında kullanılmak üzere, ‘‘setq fonksiyonu’’ kullanılarak bir değişkenen atanabilir. Bu işlemin kullanım formatı ; (GETREAL (İLETİ)) gibidir [8].

Örnek: (Setq ICCAP (GETREAL’’\n ÇEMBER İÇ ÇAPI :’’)) ; Bu fonksiyon çember iç çapının sayısal olarak girilmesini sağlar. Girilen bu değer setq fonksiyonu ile ‘‘ ICCAP’’ değişkenine atanır.

2.3.14. Setvar fonksiyonu

Bu fonksiyon, bir AutoCAD sistem değişkenini eldeki belli değerlere ayarlar ve o değeri işleme koyar. Bu fonksiyonda sistem değişkeni adının çift tırnak içinde yazılması gerekir. Bu işlevin kullanım formatı (SETVAR ‘‘CMDECHO‘‘0) şeklindedir [8].

2.3.15. Getvar

Bu fonksiyon, bir AuotCAD sistem değişkeninin değerini geri çağırır. Değişken adı çift tırnak içine alınmalıdır. Örneğin, en son belirlenmiş yuvarlama yarıçapının 0.5 birim olduğunu varsayalım,

(GETVAR ‘‘FILLETRAD’’) 0.5 olarak işleme koyar. Eğer AutoCAD tarafından bilinmieyen bir sistem değişkeninin değerini geri çağırmak için getvar kullanırsanız sıfır (nil) olarak işlem görür [8].

2.3.16. Getangle fonksiyonu

AUTOLISP’in bu fonksiyonu, bir açının radayan cinsinden girilmesini ister ve o açıyı döndürür. Getangle, ANGBASE değişkeni tarafından güncek açı olarak ayarlanan sıfır radyan yönlü açıları, açılar saatinin ters yönünde artacak şekilde ölçer. Verilen açı, güncel inşa düzlemine göre (güncel yükseklikte, UCS’nin xy düzlemi) radyan cinsinden ifade edilir.

İleti, ileti olarak görüntülenecek seçimlik bi dizgidir, nokta ise, güncel UCS’de seçimlik iki temel noktasıdır. AutoCAD güncel açı birimleri formatına girerek bir açıyı belirleyebilirsiniz. Güncel açı birimleri formatı, derece yada grad cinsinden veya başka bir birimden olabildiği halde, bu işlev açıyı her zaman radyan cinsinden işleme koyar. Ayrıca, AUTOLISP’e açıyı, grafik ekranda iki nokta işaretleyerek de gösterebiliriz. AutoCAD açıyı, görselleştirmede yardımcı olmak için, birinci noktadan güncel çapraz konuma bir esnek bant çizgisi çeker. Getangle’in seçimlik nokta argümanının, eğer belirlenmişse, bu iki noktanın ilki olduğu varsayılır; bu da bize, bir başka noktayı daha işaretleyerek AutoLISP’de açıyı göstermemize izin verir. Yazılım formatı;
GETANGLE (Nokta) (İleti)
(setq açı (getangle)) şeklindedir [8].

2.3.17. Geteorner

Geteorner işlevi, aynı getpoint gibi güncel UCS’de bir nokta döndürür. Ancak, geteorner bir noktayı temel noktası argümanı olarak ister ve kullanıcı, çaprazları ekranda hareket ettirdikçe, o noktadan bir dikdörtgen çizer. Temel nokta, güncel UCS’ye göre ifade edilir. Eğer üç temel noktası sağlarsanız, onun Z koordinatı göz ardı edilir; güncel yükseklik Z koordinatı olarak kullanılır.

2.3.18. Getdist

Bu fonksiyon, kullanıcının bir uzaklık ya da bir veya iki nokta girmesi için duraklar. Getdist, noktalar arsasındaki uzaklığı gösteren bir gerçek sayıyı işleme koyar. Uzaklığı, güncel AutoCAD uzaklık birimleri formatında girerek belirleyebiliriz. Güncel uzaklık birimleri her ne kadar feet ve ınch cinsinden (mimari) olabilir ise de, bu işlev, her zaman bir gerçek sayı olarak döndürür. Ayrıca iki noktanın , yerini de belirleyebilirsiniz ve getdist, uzaklığı görselleştirmede bize yardımcı olmak üzere, ilk noktadan güncel çapraz konumuna bir elatik bant çizgisi çizerek bu iki nokta arasındaki uzaklığı döndürür. Nokta güncel, UCS'de seçimlik iki ve üç temel noktasıdır. Eğer sağlanırsa, nokta iki noktanın ilki olarak kullanılır ve bize yalnızca ikinci nokta için ileti getirir [8].

Örnek:

(Setq uzaklık (getdist))
(Setq uzaklık (getdist '(1.0 3.5)))
(Setq uzaklık (getdist ''ne kadar uzak ''))
(Setq uzaklık (getdist ' (1.0 3.5) ''ne kadar uzak ? ''))

AUTOLISP bunlar gibi daha bir çok komutun yer aldığı bir programlama dilidir. Matematik dosyalama, AutoCAD komutları, değişken dönüşümleri, koordinat sistemleri ekran kontrolü, seçim seti fonksiyonları, varlık fonksiyonları (entity), ads aplication, hafıza düzenlemesi ve yönetimi diyalogların kontrolü ve oluşturulması, diyaloglardan veri alınması , diyaloglara veri verilmesi gibi alanlar için tasarlanmış komut veya fonksiyonlara sahiptir.

3. LİTERATÜR ÖZETİ

Yapılan literatür taramalarında mobilya ve dekorasyon alanında AUTOLISP uygulamaları üzerine çalışmaya rastlanmamıştır. Bu çalışmaya benzer çalışmalar aşağıda verilmiştir.

Erdinler (2005), Türkiye Mobilya Endüstrisinde CAD sistemlerinin kullanım etkinliğinin belirlenmesi amaçlı çalışması sonucunda; işletmelerin kullandığı programlar arasında popüler olmasından dolayı AutoCAD % 43 ile en çok kullanılan program olduğu, bu programı 3DMAX (%9.1), Photoshop (%8.1) ve ArCon (%8.1) takip ettiğini bildirmiştir. Ayrıca işletmelerin sadece % 8'inde tasarım işlemlerinde bilgisayar desteği alınmadığını, endüstriyel uygulama örneklerinden alınan veriler sonucunda CAD uygulaması ile siparişi karşılama süresi 4 gün (% 57) azalarak 7 günden 3 güne düştüğünü, CAD sisteminin doğrudan katkısı ile siparişi karşılama süresinde %57 azalma sağlandığını, CAD sistemlerinin CAM sistemleri ile entegrasyonunun sağlanmadığı durumlarda CAD sistemleri üretim miktarını doğrudan etkilemese de dolaylı olarak üretim miktarında artış gözlemlendiğini, uygulama yapılan işletmede CAD sistemi kullanımı ile üretim miktarında 4 aylık süreçte dolaylı olarak % 34 artış sağlandığı saptandığını, aynı işletmede AutoCAD kullanımı ile sipariş karşılama işlemlerinin kısa sürede karşılanmasına bağlı olarak malzeme sipariş süresi 1 aydan 1 haftaya indiğini, malzeme tedarik süresinde %90 azalma görüldüğünü bildirmiştir [1].

Taşdelen (1998), standart makine elemanlarının AutoCAD ortamında, AUTOLISP yardımıyla tasarımı konusunda yapılan çalışması sonucunda, tasarımcı AUTOLISP'in temel yapısını öğrenerek, LISP programlarını yazmaya vakıf hale gelmesi ve çalışma içerisinde kullanılan diyalog kutuları DCL sayesinde ekranda diyalog kutusu açabilecek ve program girdilerini ekrandan girme olanağına kavuşacağını belirtmiştir [10].

Bilgin (1996) yaptığı yüksek lisans tez çalışması ile; Bilgisayar Destekli Tasarımda AUTOLISP dilinin kullanımı ile faydalarını irdelemiştir. AutoCAD kullanıcılarının

AUTOLISP'ten nasıl faydalanılacağı, bu konu hakkında yeterli bilgi sahibi olmanın önemi bu çalışma içinde vurgulanmıştır. Parametrik dişli kutusu projelendirmesi esas alınarak, AUTOLISP yardımı ile uygulama yapılmıştır [11].

Katar (1995), yapmış olduğu çalışma sonucunda AUTOLISP'in parametrik olarak sürekli kullanım alanları için ideal bir program olduğunu, AutoCAD desteği de bu dili cazip hale getirdiğini ortaya koymuştur. AutoCAD ekranına aktarılan tasarımlarını ve bunların iki ya da üç boyutlu cizimlerini ve detaylarını daha yakından incelemek, üzerinden hassas ölçü alabilmek, hatta ölçekli olarak kağıda dökebilmeyi AUTOLISP yardımıyla daha kolay hale getirebildiğini ortaya koymuştur [6].

Wiebe and Summey (1995) North Carolina Universitesinde yer alan Mobilya Üretim Merkezinin katkılarıyla gerçekleştirdiği çalışmalarında Mobilya endüstrisinde Bilgisayar Destekli Tasarım ve Üretim ile ilgili mevcut eğilimleri değerlendirmişlerdir. Bu eğilimleri endüstriye uygulanan anketler, fabrika ziyaretleri, kullanılan bilgisayar sistemlerinin incelenmesi, raporlanması ve bunlarla ilgili seminerlerin hazırlanmasına dair önerilerle 6 adım da değerlendirmişlerdir [12].

Wiebe et al. (1997), mobilya endüstrisinde CAD entegrasyonu ve ürün veri yönetimi araçlarının organizasyonel değerlendirmesini yapmışlardır. Çalışmada CAD ve Ürün veri yönetimi entegrasyonunun geçişleri incelenmiştir [13].

4. MALZEME VE YÖNTEM

4.1. Malzeme

4.1.1. Kullanılan bilgisayar programı ve programlama dili

Bu çalışmada mobilya endüstrisinde tasarım amacıyla yaygın olarak tercih edilen Autodesk AutoCAD 2010 programı kullanılmıştır (30 günlük trial versiyonundan yararlanılmıştır).

Programlama dili olarak AutoCAD programı altında çalışan ve LISP kökenli olan AutoLISP dili kullanılmıştır.

AutoLISP, Davit Betz tarafından XLISP dilinden uyarlanarak ve AutoCAD programına ait birçok özellik eklenerek geliştirilmiştir. AutoLISP liste işlemek temelinde tasarlanan bir dildir. Bilgiyi ve veriyi yönetmek için programcıya birçok fonksiyon sunmaktadır. AutoLISP, kullanıcının AutoCAD'e yeni komutlar eklemesini ve kişiselleştirebilmesi sayesinde tasarımda daha yüksek verim elde etmesini mümkün kılmaktadır.

4.2. Yöntem

Bilgisayar destekli mobilya tasarımı süreci birçok adımdan oluşmaktadır. Bunlar; taslak çizim, modelleme, konstrüksiyon tasarımı, mühendislik analizler, görsel sunumlar hazırlama ve otomatik teknik çizimler hazırlama olarak sıralanabilir.

Konstrüksiyon tasarımı mobilya tasarımında en önemli adımlardan biridir. Bu aşamada parametrelerin ve işlem basamaklarının fazla olması tasarımcının fazla zaman harcamasına sebep olmaktadır. Bu nedenlerle çalışmada konstrüksiyon tasarımı aşaması problem olarak ele alınmıştır. Bu aşamadaki işlem basamaklarını ve harcanan zamanı en aza indirgenmesi hedeflenmiştir.

Mobilya konstrüksiyon teknikleri arasında parametrik modellemeye müsait olmaları ve yaygın olarak uygulanmaları dikkate alınarak çalışmada kavelalı birleştirme ve zıvanalı birleştirme teknikleri örnek olarak seçilmiştir.

4.2.1. Kavelalı birleştirme

Kavelalı birleştirme tekniği masif ve panel mobilya elemanlarının birleştirilmesinde yaygın olarak kullanılan bir tekniktir. Bilgisayar destekli olarak bu birleştirme tekniğinin modellenmesini kolaylaştırmak amacıyla hazırlanan AutoLISP programında işlem;

- Birleştirme yapılacak elamanların modellenmesi,
- Kavela sayısı ve çapının kullanıcı tarafından belirlenmesi,
- Elemanların seçimi,
- Yüzeylerin seçimi,

adımlarından oluşmaktadır. Program yazıldıktan sonra AutoCAD programına nasıl yükleneceği anlatılmış, ayrıca örnek bir uygulama yapılarak işlem basamakları ayrıntılı şekilde açıklanmıştır. Örnek çalışmada endüstride yaygın olarak uygulaması dikkate alınarak panel elemanlar ve 8mm ve 10mm çapında, 35mm uzunluğunda kavelalar kullanılmıştır.

4.2.2. Zıvanalı birleştirme

Zıvanalı birleştirme tekniği masif mobilya elemanlarının birleştirilmesinde yaygın olarak kullanılan bir tekniktir. Bilgisayar destekli olarak bu birleştirme tekniğinin modellenmesini kolaylaştırmak amacıyla hazırlanan AutoLISP programında işlem;

- Birleştirme yapılacak elamanların modellenmesi,
- Zıvana ölçülerinin belirlenmesi,
- Elemanların seçimi,
- Yüzeylerin seçimi,
- Yuvarlatılacak kenarların seçimi,

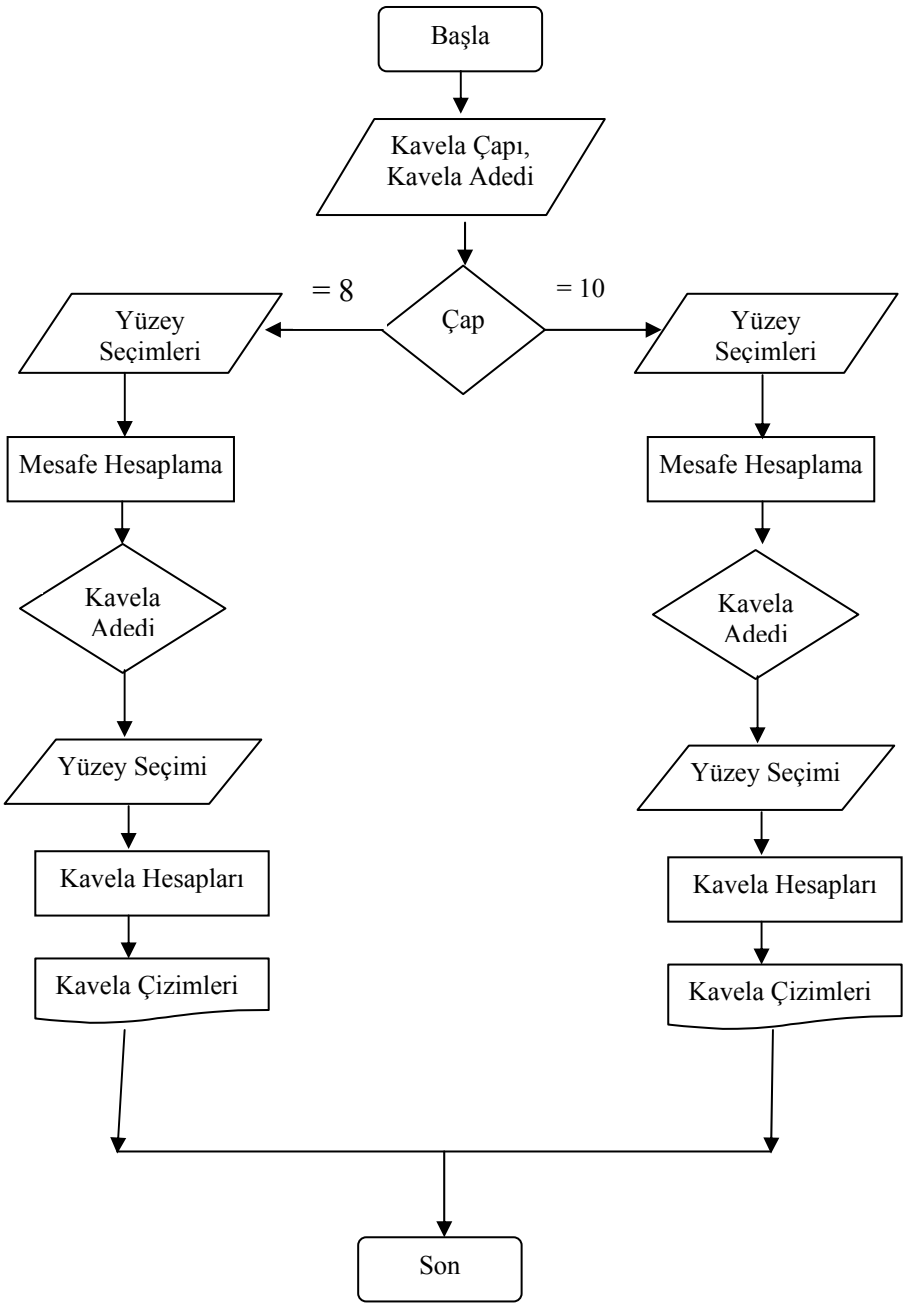
adımlarından oluşmaktadır. Program yazıldıktan sonra AutoCAD programına nasıl yükleneceği anlatılmış, ayrıca örnek bir uygulama yapılarak işlem basamakları

ayrıntılı şekilde açıklanmıştır. Örnek çalışma endüstride yaygın olarak uygulaması dikkate alınarak tek zıvanalı ayak-kayıt birleştirme üzerinde yapılmıştır.

5. MOBİLYA TASARIMINDA AUTOLISP UYGULAMALARI

5.1. Kavelalı Birleştirme

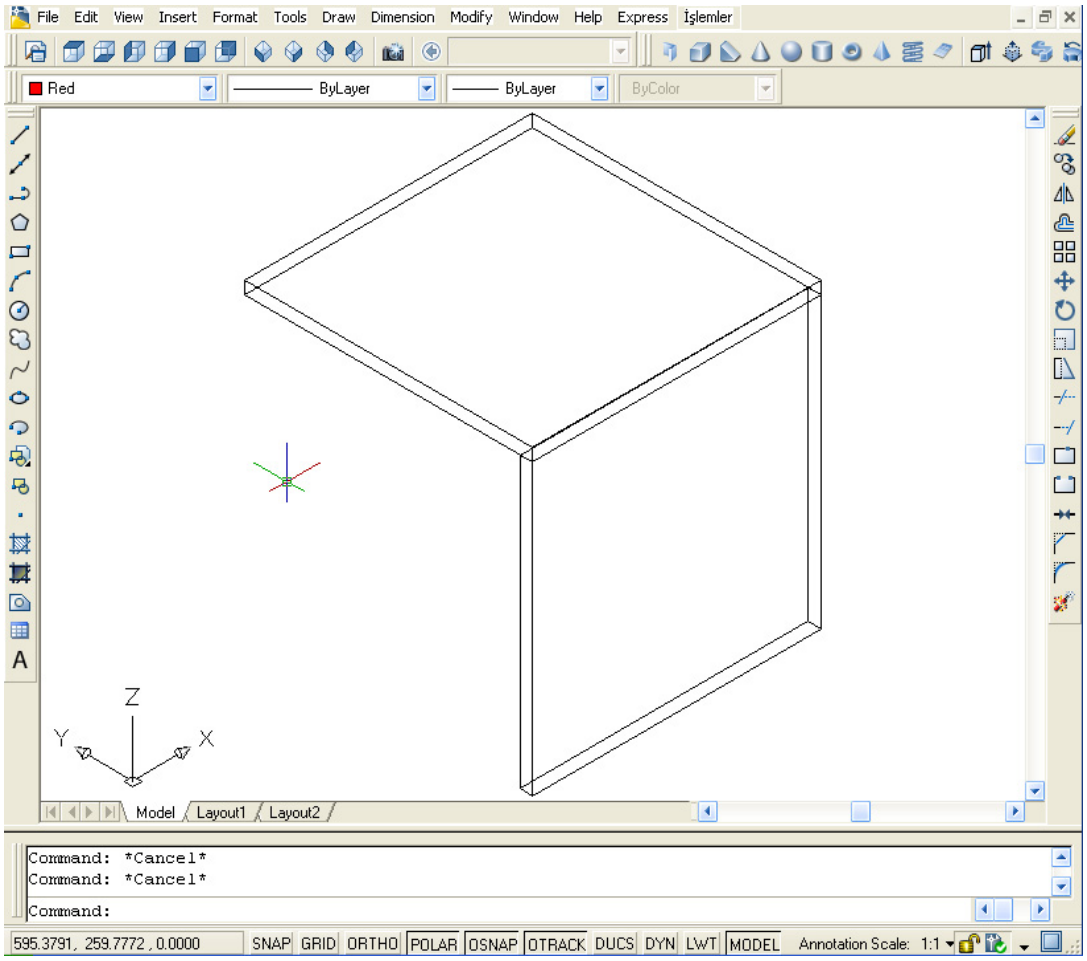
Kavelalı birleştirme lisp programında gerçekleşen işlem basamakları Şekil 5.1'de verilmiştir.



Şekil 5.1. Kavelalı birleştirme algoritması

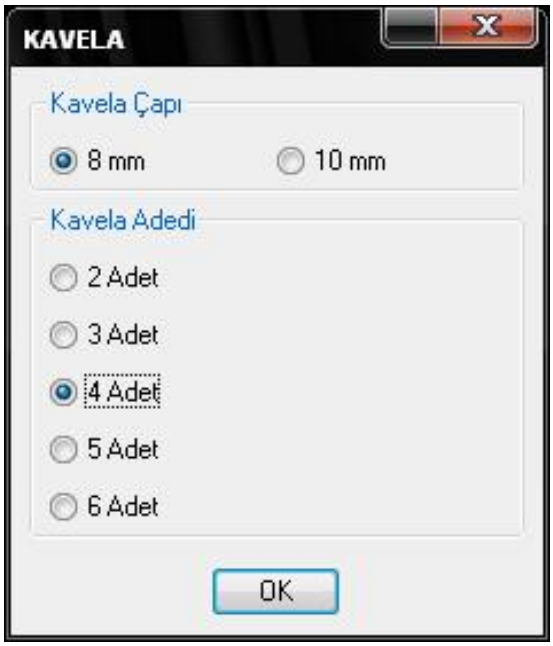
Kavelalı birleştirme tekniği uygulaması çalıştırılmadan önce kullanıcının yapması gereken birkaç işlem bulunmaktadır. Bunlardan birincisi model ekranına birleştirme yapacağı parçaların modellenmesidir. Kullanıcı model ekranına birleştirme yapacağı parçaları çizdikten sonra lisp dosyasını AutoCAD ortamına yüklemesi gerekmektedir. AutoCAD menüleri yardımı ile Tools menüsünden Load Application seçeneği seçilerek kaynak dosya işaretlenerek (Örneğin : Kavela.lsp) load tuşuna basılır ve bu sayede EK-1'de verilen Lisp uygulama dosyası yüklenmiş olur.

Kullanıcı tarafından çizilmiş bir mobilya üst ve yan tablasına Şekil 5.2'de bu birleştirme tekniğinin uygulanması anlatılmıştır.



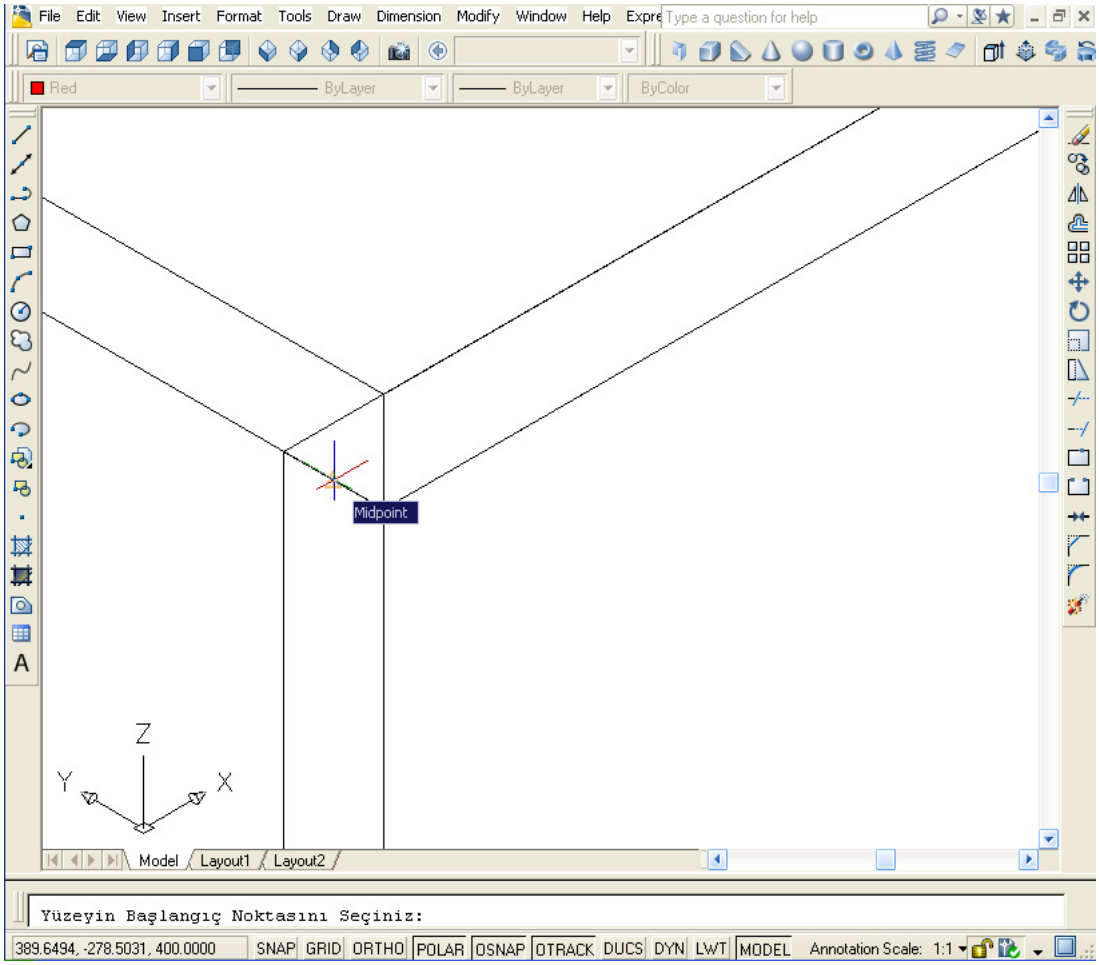
Şekil 5.2. Örnek bir kullanıcı çizimi

Menüden veya komut satırından kavela komutu girilerek uygulama çalıştırılır. Komut çalıştığında ekrana kavelanın çapı ve adedinin girileceği bir pencere gelecektir (Şekil 5.3).



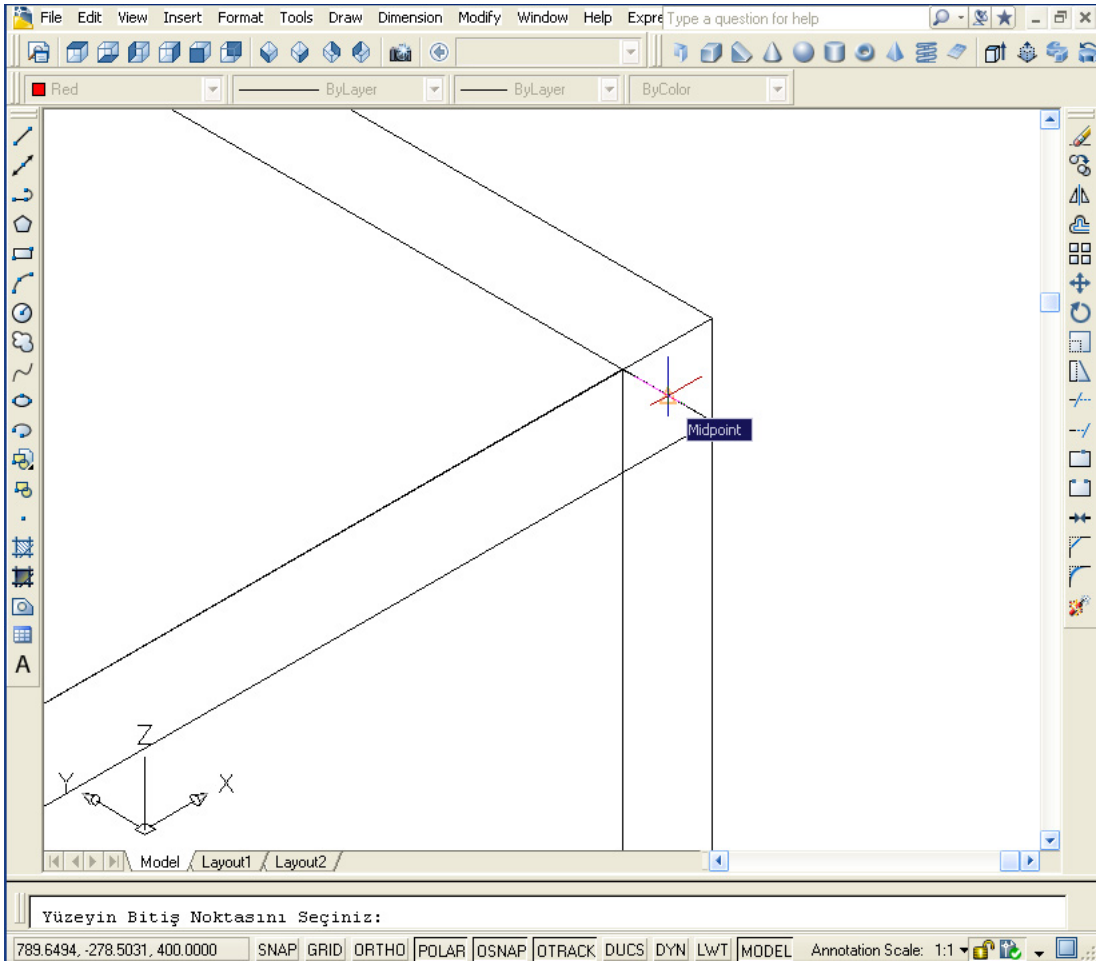
Şekil 5.3. Kavela diyalog penceresi

Bu pencereden kavela çapı ve kavela ededi seçilerek "OK" butonuna basılır. Daha sonra Şekil 5.4'de gösterildiği gibi kullanıcıdan kavela yerleştirilecek yüzeyin başlangıç noktasının seçilmesi istenecektir.



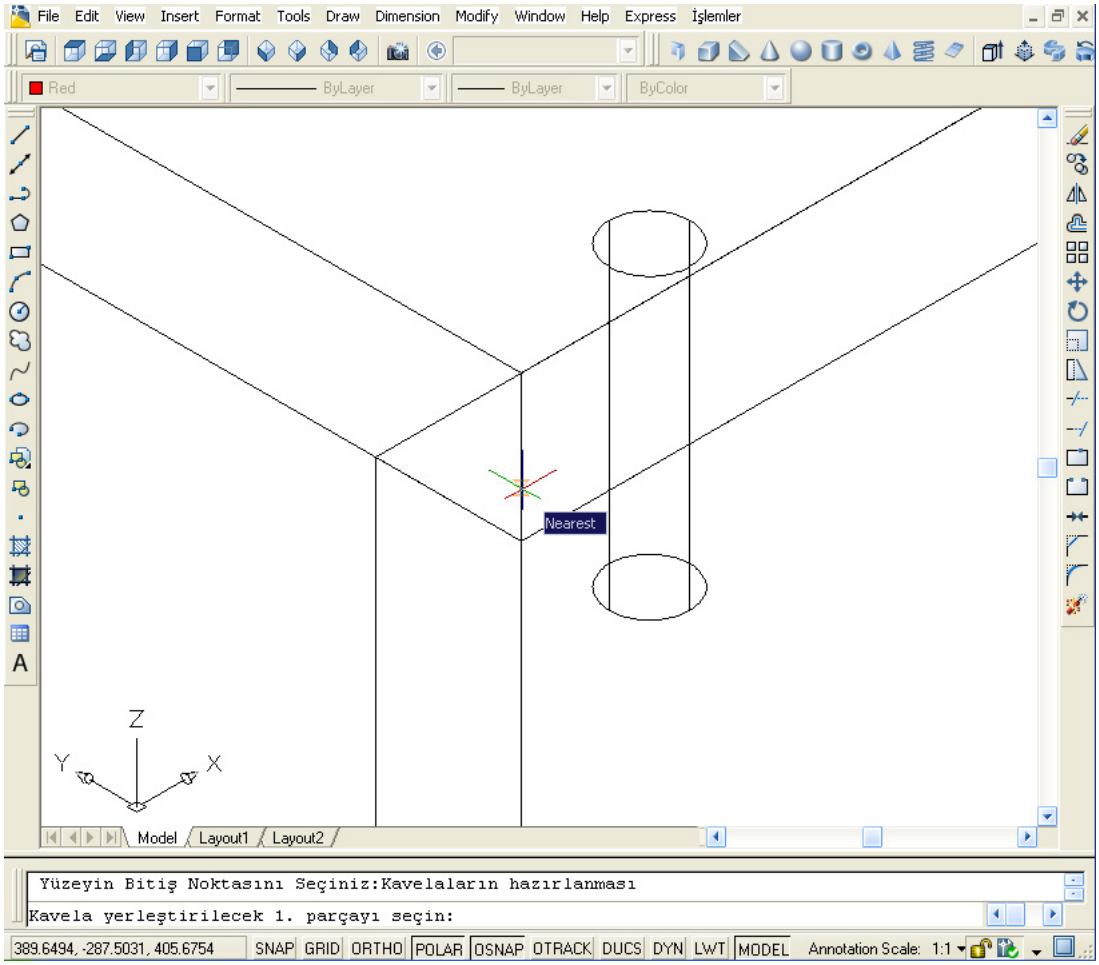
Şekil 5.4. Yüzey başlangıç noktası seçimi

Şekil 5.4'de görüldüğü gibi yüzeyin başlangıç noktasına zoom yapılarak başlangıç noktasının ortası seçilir, seçimden hemen sonra kullanıcıdan yüzeyin bitiş noktasını seçmesi istenecektir (Şekil 5.4.). Bu şekilde program başlangıç ve bitiş noktaları arasındaki uzunluğu hesaplayarak belirlenen kavela sayısını bu aralığa yerleştirecektir.



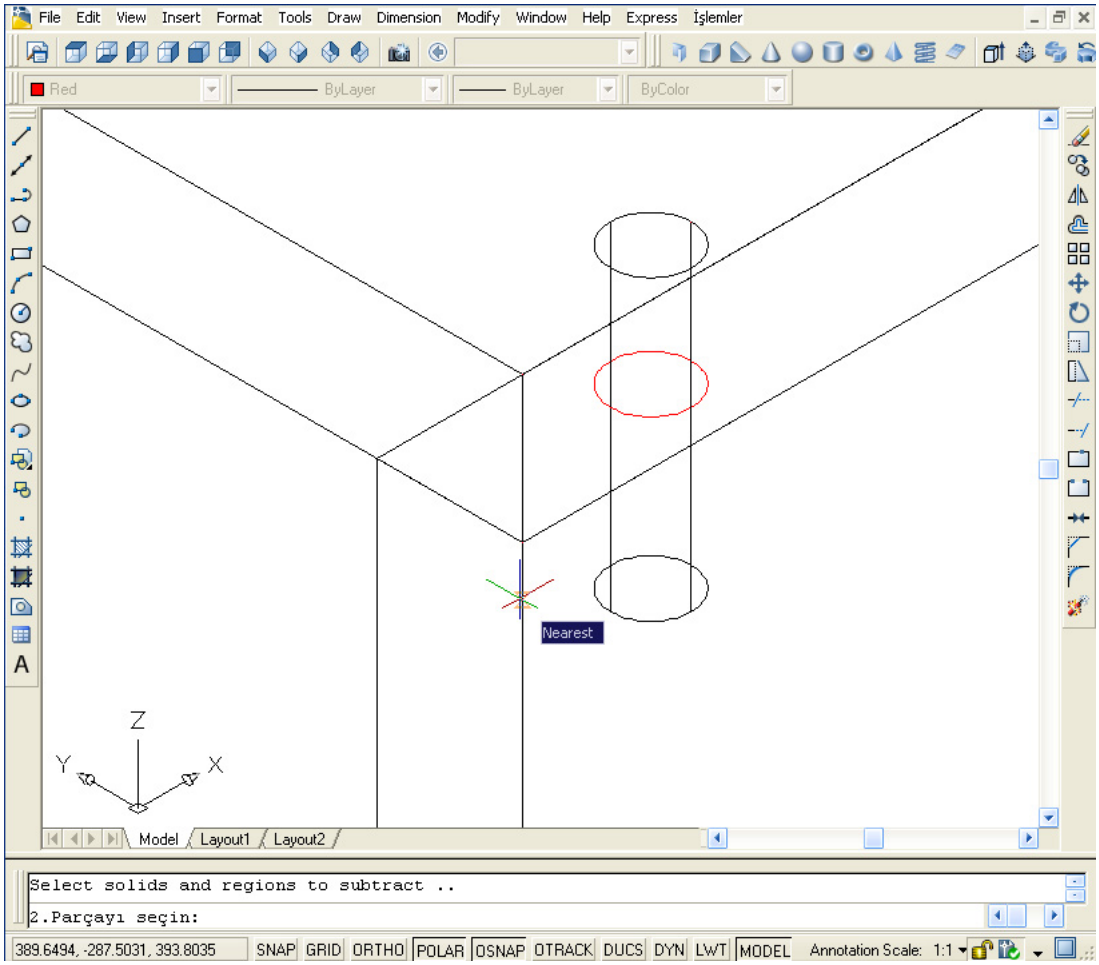
Şekil 5.5. Yüzeyin bitiş noktasının seçimi

Yüzeyin başlangıç ve bitiş noktaları seçildikten sonra program kullanıcıdan kavela yerleştirilecek üst ve yan tablayı seçmesini isteyecektir. Seçim işlemi Şekil 5.5'de görüldüğü gibi yüzeyin başlangıç noktasına zoom yapılarak üst tabla seçilir daha sonra program bizden 2. parça olan yan tablayı seçmemizi isteyecektir (Şekil 5.6).



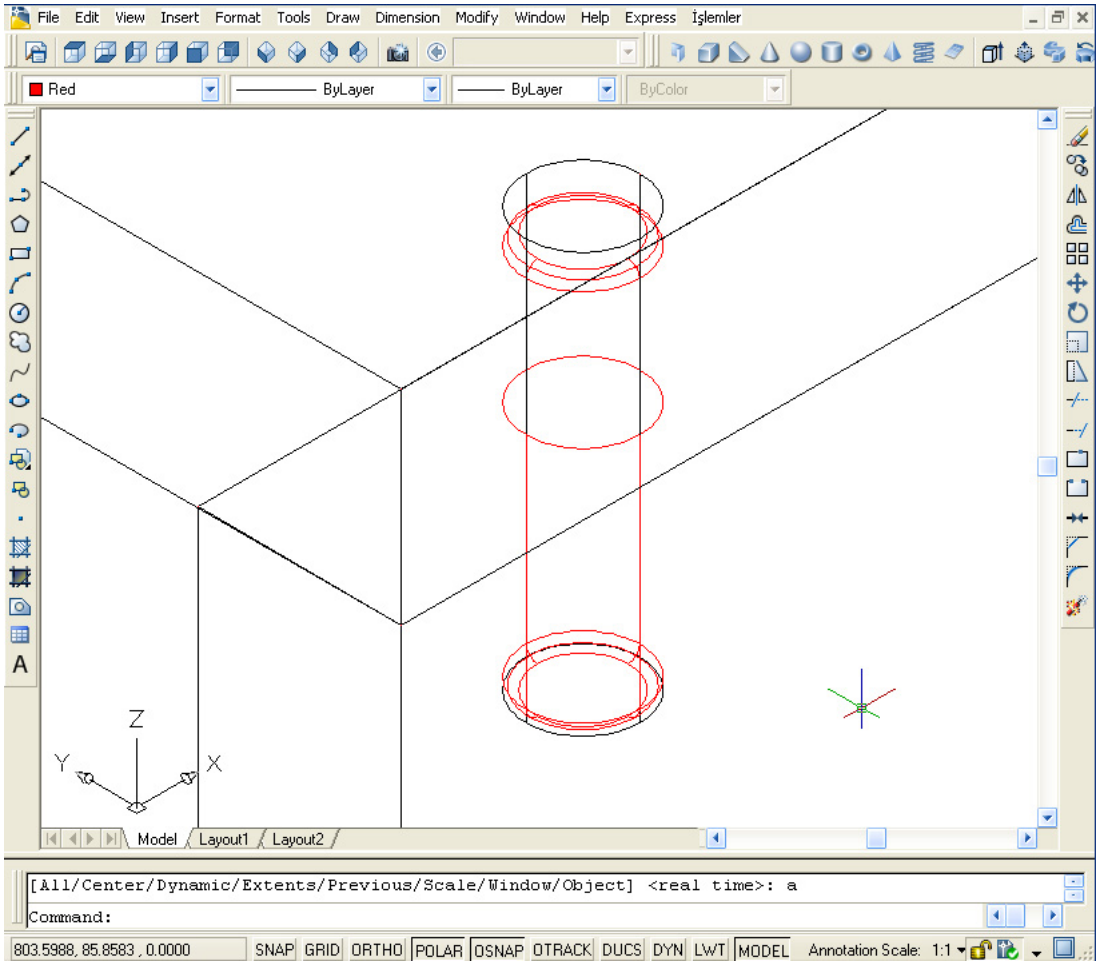
Şekil 5.6. Üst ve yan tablanın seçimi

Şekil 5.6'da görüldüğü gibi komut satırında kullanıcıdan kavela yerleştirilecek 1. parçayı seçmesi için uyarı çıkacaktır, kullanıcı üst tablayı seçtikten sonra komut satırında 2. parçayı yani yan tablayı seçmesi için yeniden bir uyarı çıkacaktır (Şekil 5.7.).



Şekil 5.7. Yan tabla (2. parçanın) seçilmesi

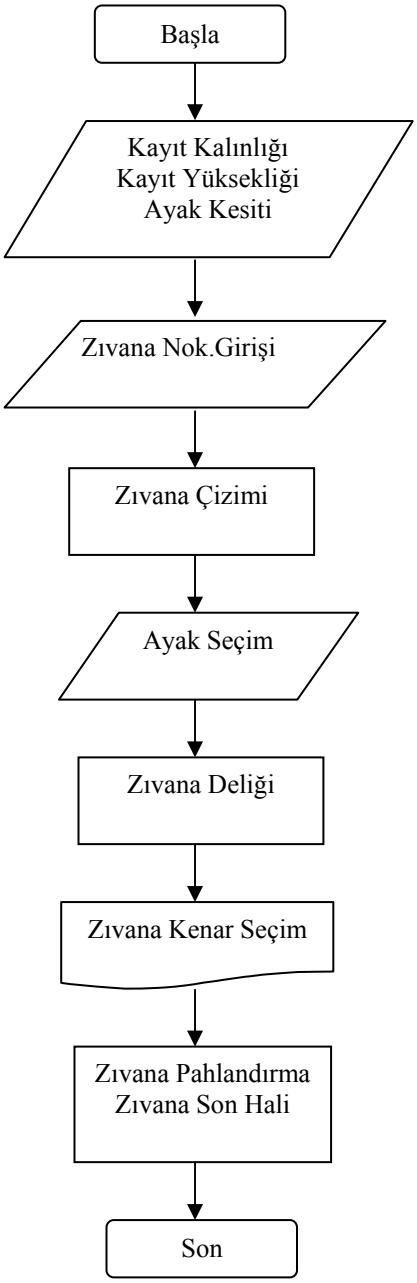
Kullanıcı yan tablayı Şekil 5.6'da görüldüğü gibi seçerek kavelaların yerleştirilmesindeki son adımı gerçekleştirir. Seçim işleminden sonra program bu iki parçaya Şekil 5.7. ve Şekil 5.8'de görüldüğü gibi kavela deliklerini açacak ve modellediği kavelaları yerleştirerek birleştirme işlemini tamamlayacaktır.



Şekil 5.8. Kavelaların yerleştirilmesi

5.2. Zıvanalı Birleştirme

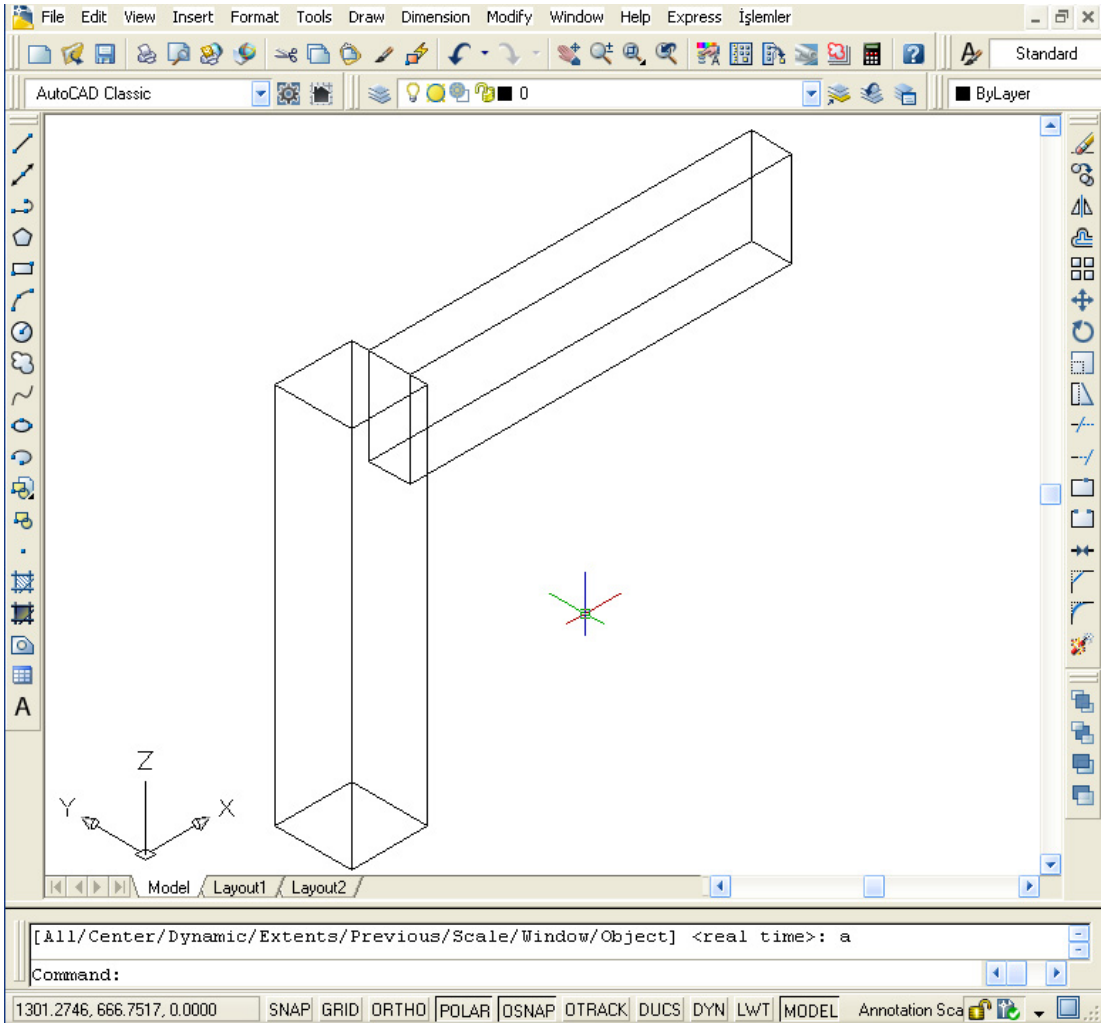
Zıvanalı birleştirme lisp programında gerçekleşen işlem basamakları aşağıda verilmiştir (Şekil 5.9).



Şekil 5.9. Zıvana birleştirme algoritması

Kavelalı birleştirme tekniğinde olduğu gibi kullanıcı birleştirme yapmak istediği parçaları modellemelidir. Elemanların modellenmesinden sonra lisp dosyasını AutoCAD ortamına yüklemesi gerekmektedir. AutoCAD menüleri yardımı ile Tools menüsünden Load Application seçeneği seçilerek kaynak dosya işaretlenerek (Örneğin : Zıvana.lsp) load tuşuna basılır ve bu sayede EK-2'de verilen Lisp uygulaması dosya içerisine yüklenmiş olur.

Kullanıcı tarafından modellenen örnek bir ayak kayıt birleştirme elemanları Şekil 5.10'da verilmiştir.

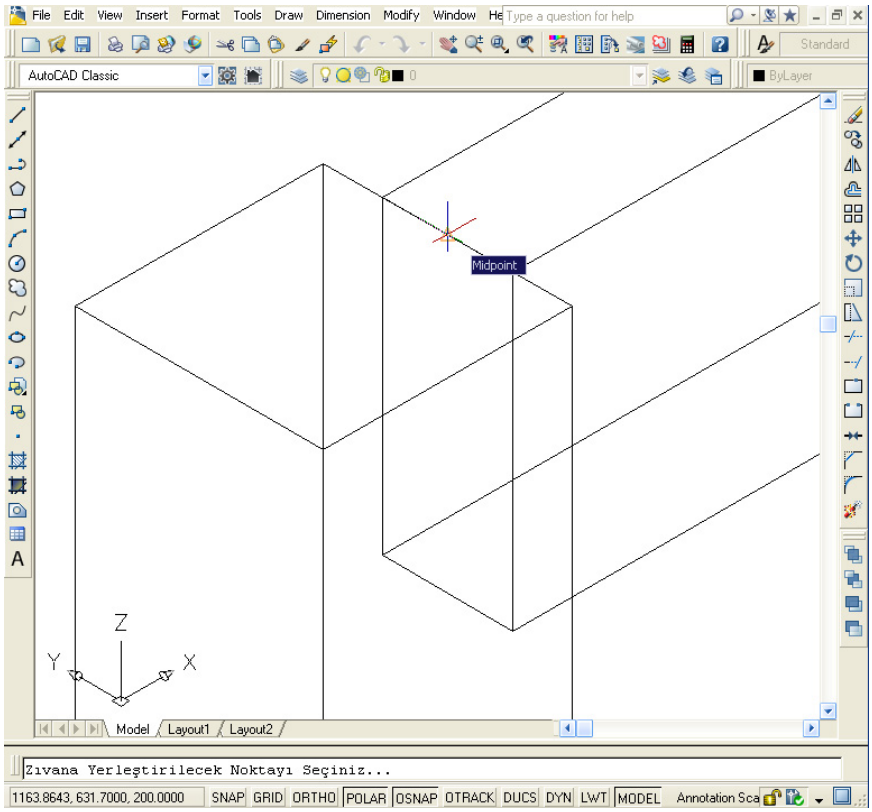


Şekil 5.10. Örnek bir kullanıcı modelleri

Komut satırından veya menüden zıvana komutu girilerek uygulama çalıştırılır. Komut çalıştığında ekrana Şekil 5.11'de görüldüğü gibi kullanıcıdan zıvanalı birleştirmeyle ilgi parametrelerin girilmesi istenecektir.

Şekil 5.11. Zıvana diyalog penceresi

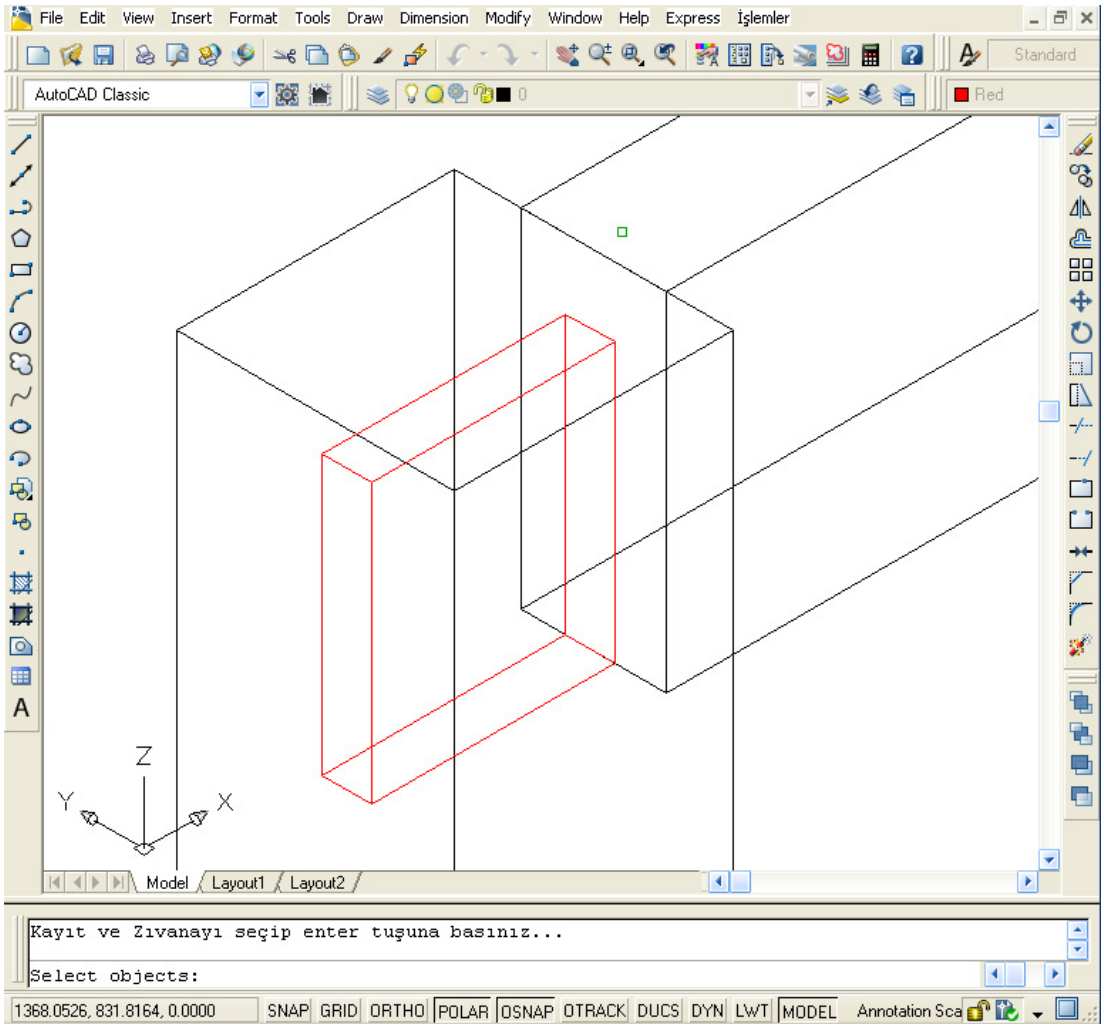
Örnek uygulamada kayıt kalınlığı 21mm, kayıt yüksekliği 50mm ve ayak kesiti ise 40mm olarak tanımlanmıştır. Bundan sonraki aşamada program kullanıcıya zıvana yerleştirecek noktayı soracaktır (Şekil 5.12).



Şekil 5.12. Zıvana yerleştirilecek noktanın seçilmesi

Zıvana yerleştirilecek nokta seçiminde ekran seçim yapılacak bölgeye zoom yapılarak yanlış nokta seçme riski azaltılmış olacaktır. Seçim ayak kesitinin ve kayıtın birleşme noktalarının üst yüzeyinde ve orta nokta olan (Şekil 5.12.'de görüldüğü gibi) midpoint seçeneği çıktığında yapılmalıdır. Program bu sayede seçilen bu orta noktayı referans alarak zıvana modellemesini gerçekleştirecektir.

Referans nokta seçildikten hemen sonra program belirlenen standartlarda Şekil 5.13'de görüldüğü gibi parametreleri önceden girilen zıvananın modellemesini gerçekleştirecektir.

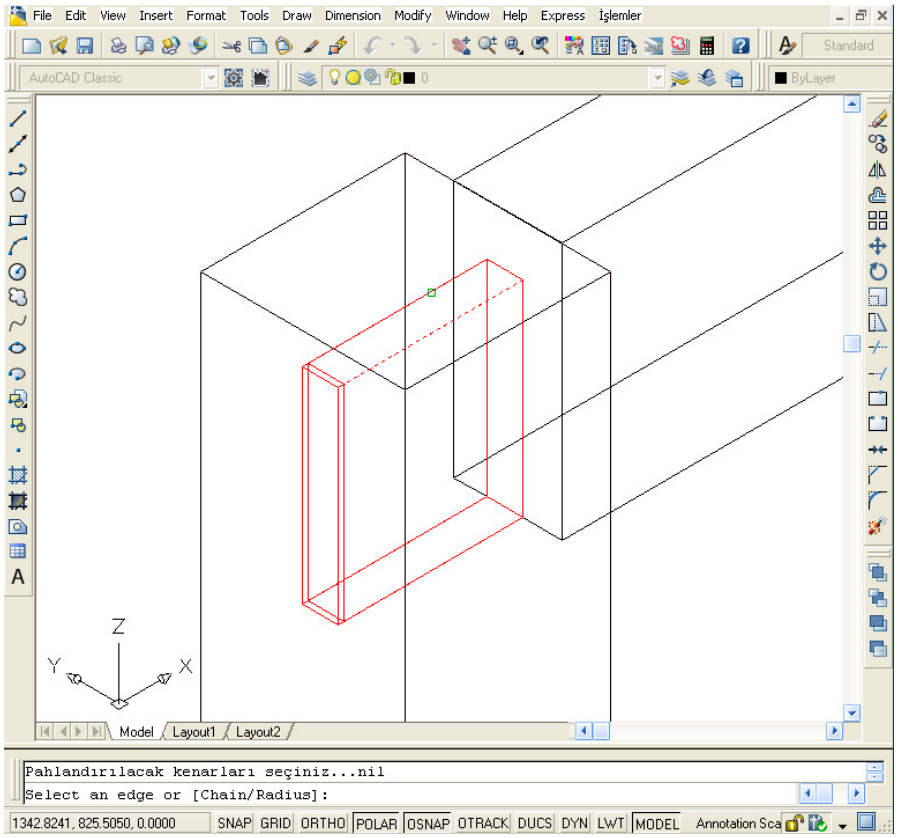


Şekil 5.13. Zıvananın modellenmesi

Bu işlem bittikten sonra komut satırında “Kayıt ve Zıvanayı seçip Enter tuşuna basınız…” açıklayıcı mesajı belirecektir. Komut satırında ise “select obje” yani kullanıcıdan nesneleri seçmesi beklenecektir, kullanıcı kayıtı ve çizilen zıvanayı seçtikten sonra “enter” tuşuna basarak kayıt ve zıvanayı birbiri ile birleştirmiş olacaktır. Bu işlemler sonucunda kayıta zıvana eklenmiş duruma gelecektir.

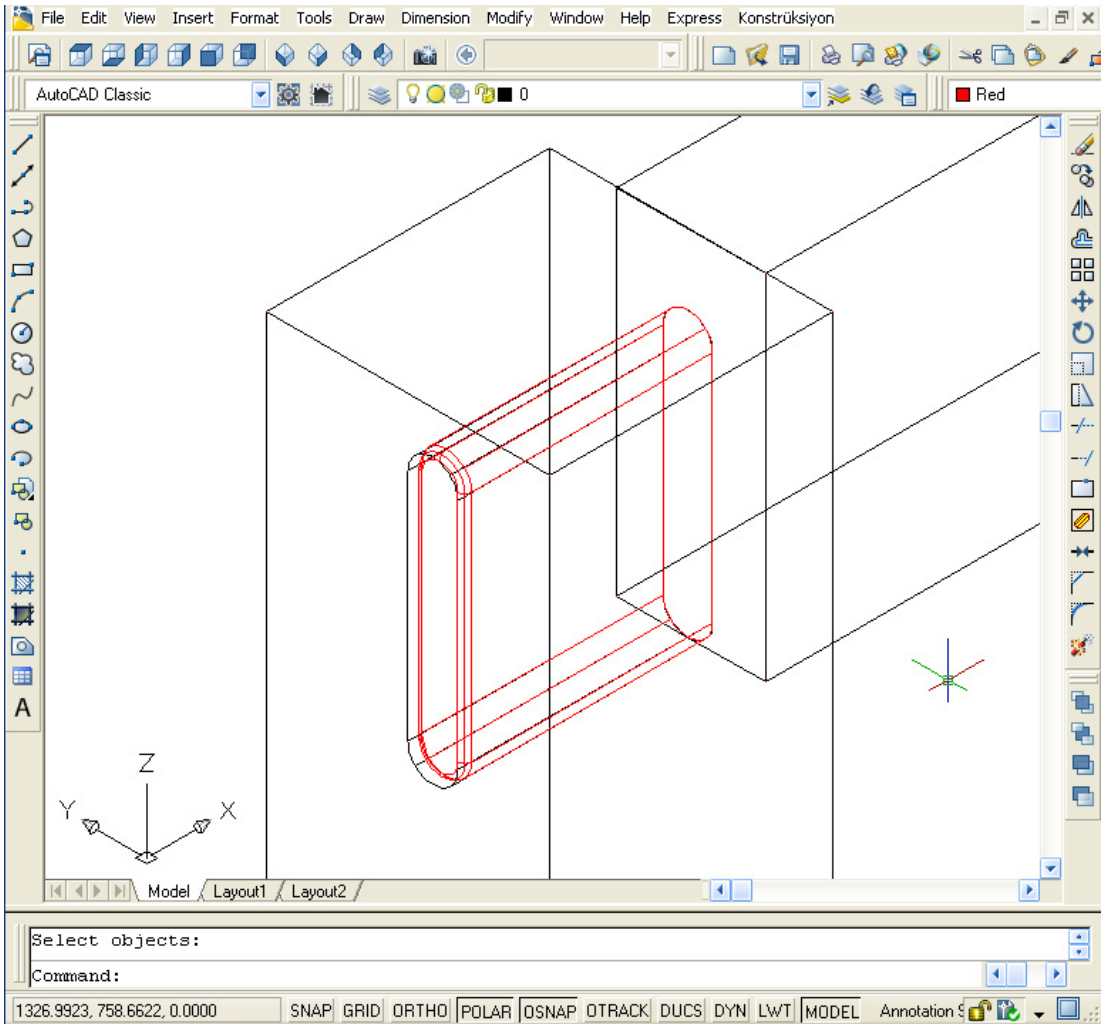
Tek parça haline getirilen kayıttan sonra kayıtın karşılığı olan deliğin açılması için kullanıcıdan diğer elemanın seçilmesi istenecektir. Kullanıcı ayak kesitinin her hangi bir noktasından seçerek seçim işlemini gerçekleştirir. Seçim sonucunda ayak kesitine zıvananın karşılığı olan delik program tarafından hesaplanarak ayak kesit üzerine yerleştirilir ve ayak kesitten çıkartılarak zıvananın karşılığı olan kısım meydana getirilmiş olur.

İşlemler gerçekleştirildikten sonraki aşamada zıvana kenarlarının yuvarlatılması aşamasıdır. Bu aşamada yuvarlatılacak kenarların seçimi istenecektir (Şekil 5.14).



Şekil 5.14. Zıvananın kenarlarının seçilmesi

Komut satırında “Pahlandırılacak kenarları seçiniz…” adlı bir bilgilendirme mesajı görülecek, bir alttaki satırda ise Select an edge or [Chain/Radius] komutu belirecektir. Yuvarlatılacak kenarlar seçildikten sonra enter tuşuna basılır ve Şekil 5.15’deki gibi zıvananın kenarları pahlandırılarak birleştirme işlemi tamamlanacaktır.



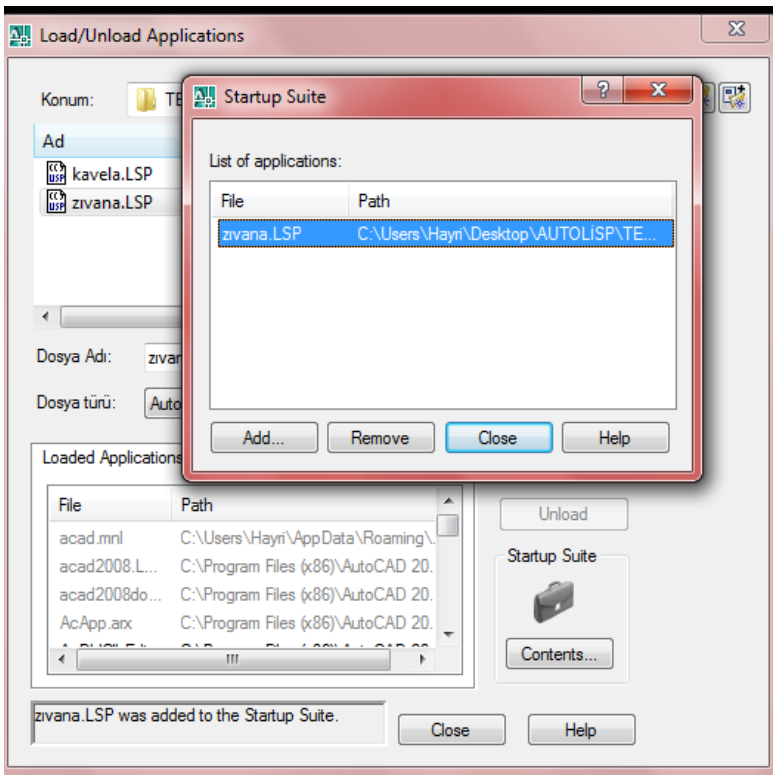
Şekil 5.15. Zıvananın birleştirme işleminin tamamlanması

5.3. Lisp Dosyalarının AutoCAD Menülerine Eklenmesi

Kullanıma hazır hale getirilen lisp dosyalarını kullanıcının daha kolay yükleyebilmesi ve erişebilmesi için AutoCAD programı kullanıcıya bazı kolaylıklar

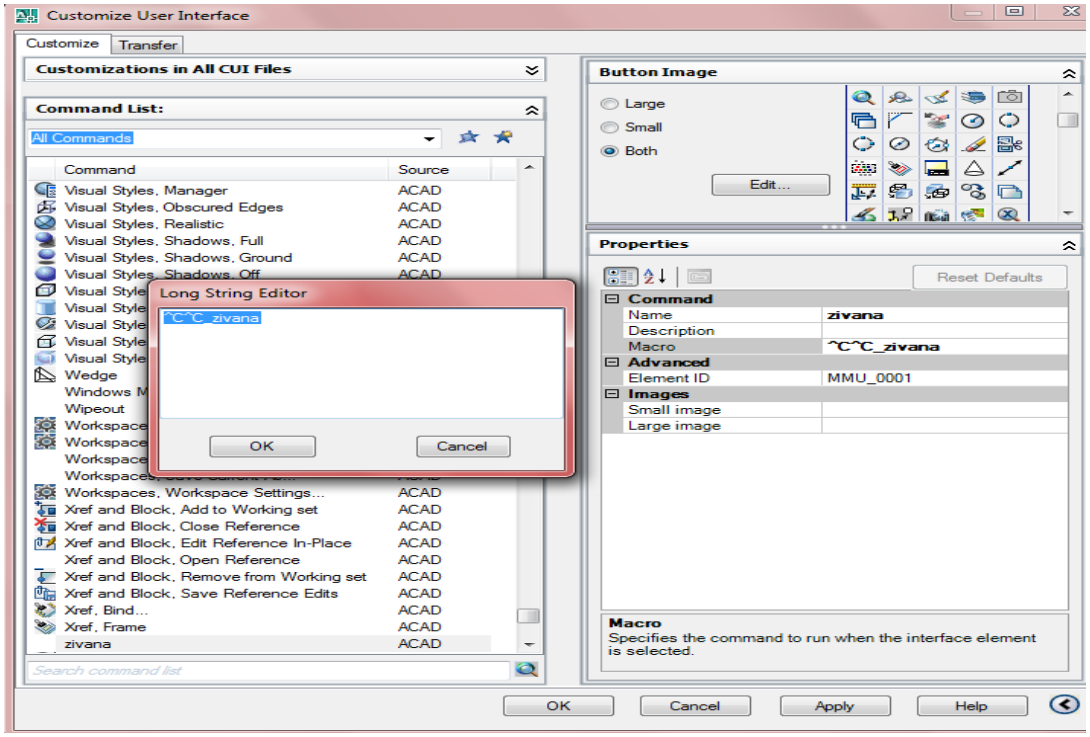
sunmaktadır. Aşağıda anlatılacak olan yöntemler izlenerek bu işlemler gerçekleştirilmektedir.

Lisp dosyalarının her defasında otomatik olarak yüklenmesi için bir defaya mahsus olarak Tools menüsünden Load Application seçeneği seçilir (Şekil 5.16).



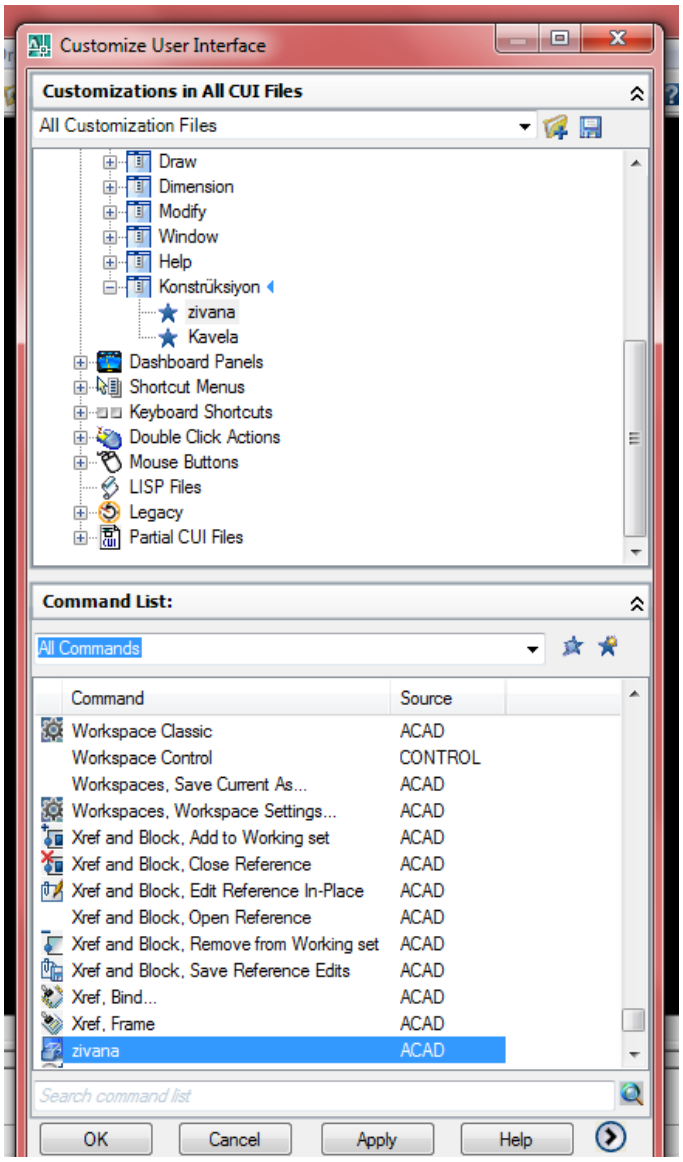
Şekil 5.16. Lisp dosyasının yüklenmesi

Seçenek seçildikten sonra ekrana gelen (Şekil 5.16.) diyalog penceresinden Contents kutusu işaretlenir. Ekrana çıkan Startup Suite adlı diyalog penceresinden Add kutucuğu seçilerek yüklemek istenilen lisp dosyası gerekli konumdan seçilerek tamam butonuna basılır. Bu işlemler gerçekleştirildikten sonra ok işaretiyle gösterildiği gibi (örneğin zıvana.LSP) seçilen lisp dosyası yüklenmiş olur. Ekrandan çıkmak için close tuşları kullanılır ve bu sayede seçilen lisp dosyası AutoCAD her açıldığında yüklenmiş olur. Yüklenmiş olan lisp dosyalarını AutoCAD'in menülerine taşımak için Şekil 5.17.'deki yolları izlemek gerekmektedir.



Şekil 5.17. Komutların eklenmesi

View menüsünden Toolbars seçeneği seçildiğinde ekrana Şekil 5.17.'deki pencere gelir. İlk olarak 1 numarası ile gösterilen create a new command kutusu tıklandığında listede Command1 adında bir yazı belirecektir, yapılacak işlem bu komutu kavela.lsp ve zıvana.lsp dosyasına uyarlamaktır. Ok işareti ile gösterilen yere tıklandığı zaman sağ tarafa ekran genişleyecek ve komutun özellikleri açılacaktır. Command başlığı altındaki "Name" yazan yazan kısmın karşısındaki kutuya tıklanarak buradaki yazı "Kavela" olarak değiştirilir. Macro kısmı karşısında bulunan 3 numarası ile gösterilen kısma tıklanarak 4 numaralı bölümdeki "^C^C_kavela" yazı yazılarak OK kutucuğuna tıklanır. Buradaki son işlem için 6 numaralı kısımda gözüken OK kutucuğuna basılarak pencereden çıkılır. Aynı işlemler zıvana kısa yolu içinde uygulanır (Şekil 5.18).



Şekil 5.18. Menülerin eklenmesi

View menüsünden tekrar Toolbars seçeneği seçildiğinde ekrana Şekil 5.18.'deki pencere açılır. Ekranın sağ üst köşesindeki ok işareti ile gösterilen kısım tıklanarak ekran genişletilir, menü seçeneğinin üzerinde fare ile sağ tuş tıklanarak New Menu Seçeneği seçilir (1 numarası ile gösterilen kısım). Menülerin hemen alt kısmında çıkan kutucuğa konstrüksiyon başlığı yazılarak enter tuşuna basılır. Alt kısımdaki Command List bölümündeki Command başlığı altında 2 numarası ile gösterilen kısımdan kavela komutu tıklanıp sol fare tuşu basılı tutularak yukarıdaki konstrüksiyon menüsü içerisine sürüklenip bırakılır (3 numaralı kısım). Aynı

işlemler zıvana komutu içinde yapılarak OK tuşuna basılıp çıkılır. Şekil 5.19'da Konstrüksiyon menüsü görülmektedir.

Şekil 5.19. Konstrüksiyon menüsü

Yapılmak istenilen işlem kullanıcı tarafından bu menüden seçildiğinde bağlı olduğu lisp dosyası çalışarak işlemlerini gerçekleştirebilecektir.

6. SONUÇ VE ÖNERİLER

Bu çalışmada bilgisayar destekli mobilya tasarımında AutoLISP kullanımına yönelik uygulamalar yapılmıştır. Bu maksatla mobilya tasarımında tasarımcının en fazla zaman harcadığı konstrüksiyon problemlerinin AutoLISP ile çözümlenmesi hedeflenmiştir. Mobilya konstrüksiyonlarında yaygın olarak uygulanan kavelalı birleştirme ve zıvanalı birleştirme teknikleri örnek olarak seçilmişlerdir. Bu maksatla yazılan lisp programlarının nasıl çalıştığı örnekler üzerinde gösterilmiş, programların AutoCAD ortamında nasıl kullanılacağı anlatılmıştır.

AutoCAD programı tasarımcıya ilave programlar ve menüler üzerinde değişiklik yapma imkanı vermektedir. Bu özelliğinden dolayı ne kadar gelişmeye elverişli bir program olduğu anlaşılmaktadır. Kendi içinde mevcut bulunan özelliklerden bir çoğu yine kullanıcı tarafından ilave edilmiştir. Örneğin bir mimar kendi alanı ile ilgili çizim ve sembolleri üretip gerekli programları ilave ettikten sonra AutoCAD programı, mimari çizimler alanlında daha verimli hale gelmiş olacaktır. Diğer yandan bir mobilya ve dekorasyoncu kendi alanına yönelik ilave LISP programları yazması AutoCAD'i bu alanda daha güçlü yapacaktır. Aynı şekilde tüm mühendislik alanlarında da uyarlamalar yapılabilir.

Bu çalışmada AutoLISP hakkında kısaca bilgi verilmiş ve AUTOLISP'in temel yapısı anlatılmıştır. Mobilya tasarımcıları bu sayede AutoLISP'in temel yapısını öğrenerek LISP programları yazmaya vakıf olacaktır.

Autolisp kullanımının özellikle konstrüksiyon tasarımı gibi önceden belli parametrik değerleri bilinen (kavelalı birleştirme, zıvanalı birleştirme vb.) tasarımlarda standartlaşma ve zaman açısından son derece önemli faydalar kazandıracağı düşünülmektedir. Bu bakımdan bilgisayar destekli mobilya tasarımcılarının Autolisp dilinin temel yapısını öğrendikten sonra yazacakları modül programlar sayesinde AutoCAD'i çok daha verimli kullanacakları söylenebilir.

Çalışma sonunda; AutoLISP'in mobilya tasarımında kullanımıyla birçok avantaj elde edileceği düşünülmektedir. Bunlar;

1. AutoCAD komutlarının tümünü kullanabilmesi açısından diğer programlama dillerine göre çizim, boyutlandırma ve ölçeklendirme sistemlerinde avantaj sağlamaktadır. Çok basit komutlarla geometrik yapıların çizimleri çabuk ve düzgün yapılabilmektedir. Mobilya tasarımında da bu bakımdan önemli avantajlar sağlandığı görülmüştür.

2. Parametrik olarak yapılan tasarımların çizimleri uzun zaman alırken; (AutoCAD ortamında) AutoLISP programı sayesinde bu çizimleri kolay ve çabuk yapmak mümkün olmaktadır. Farklı veriler için, sürekli yapılan işlemlerde, yani parametrik tasarımlarda normal çizime (AutoCAD ortamında gerçekleştirilen) göre zaman kaybını ortadan kaldırmaktadır. Özellikle kavelalı birleştirme gibi aynı işlemin çok sayıda tekrarlandığı durumlarda zaman kazanmak daha önem arzetmektedir.

3. AutoLISP programlama dili ile AutoCAD komutları arasında yer almayan ve kullanıcının sık sık kullandığı bazı hesaplama ve çizimlerin, hatta düzeltmelerin pratik kullanımlı bir AutoCAD komutu haline getirilebilmesi mümkündür. (çalışmadaki kavela ve zıvana komutları gibi). Kısaca yeni AutoCAD komutları oluşturulabilir. Sektörde faaliyet gösteren işletmelerin bu dil sayesinde özel imalatlarına yönelik programlar yazarak AutoCAD programını özelleştirmeleri firmalara tasarım aşamasında önemli esneklikler kazandıracaktır.

4. Çalışmada da görüldüğü gibi AutoLISP ile hazırlanmış bir programın kullanımı daha kolaydır. Bunun için işletmelerin bu yolla AutoCAD kullanımını daha kolay hale getirerek işletmedeki bütün personelin programı kullanabilir olmasını sağlamaları mümkündür.

5. AutoLISP komutları diğer programlama dillerinde kullanılan komutlarla benzerlik gösterir. Örneğin print, if, open, close gibi bir çok komut, mantıksal

ifadeler, komut diziliş mantığı benzerliklerden bir kaçıdır. Bu yüzden, bu benzer dillerden birini bilen biri AutoLISP'i kolayca kullanarak programlar yazabilir. Ekteki programlar incelendiğinde programların kolay anlaşılabilir oldukları görülecektir.

6. AutoLISP ile programlanan kullanıcı diyolag pencereleri (DCL) yardımıyla veri girişleri kolay bir biçimde, veri çıkışları estetik bir şekilde gerçekleştirilebilmektedir. Kavelalı ve zıvanalı birleştirme uygulamasında da görüldüğü gibi parametrik değerler menülere taşınarak çok hızlı ve kolay bir şekilde veri girişi mümkün hale gelmektedir.

Bu çalışmanın devamı olarak diğer birleştirme teknikleri için yazılacak programlar AutoLISP'in tanınmasını, kolay uygulanabilirliğinin keşfedilmesini ve sonuç olarak mobilya tasarımlarında yaygın olarak kullanılmasını sağlayacağı düşünülmektedir.

KAYNAKLAR

1. Erdinler, G.S., "Cad Sistemleri ve Türkiye Mobilya Endüstrisinde Uygulanma Etkinliğinin Analizi", Doktora Tezi***, İstanbul Üniversitesi Fen Bilimleri Enstitüsü*****,** İstanbul, 1-31 (2005).

2. Erecek, A., "BDT/BDÜ Uygulamaları ve Bir Yazılım Sisteminin Tanıtımı", Yüksek Lisans Tezi***, Erciyes Üniversitesi Sosyal Bilimler Enstitüsü,*** Kayseri, 1-7 (1993).

3. Kurtoğlu, A., Koç, H. K., Aksu, B., "Avrupa Topluluğu İle Gümrük Birliği Sonrası Türkiye Mobilya Sanayinin Rekabet Düzeyi**",** ***I. Ulusal Mobilya Kongresi Bildiri Kitabı***, Ankara,17-18 Kasım 1-10 (1997).

4. Kurtoğlu, A., Koç, H. K., Aksu, B., "Türkiye Ahşap Mobilya Endüstrisinin Dış Ticaret Analizi", ***I. Ulusal Mobilya Kongresi Bildiri Kitabı,*** Ankara, 17-18 Kasım- 1-12 (1997).

5. Koç, H. K., Aksu, B., "Küçük Ölçekli Bir Mobilya İşletmesinde Üretim Sürecinin Analizi", ***İstanbul Üniversitesi Orman Fakültesi Dergisi***, İstanbul, 1-14 (1995).

6. Katar, T., "Bilgisayar Destekli Makine Tasarımında AUTOLISP Uygulamaları", Yüksek Lisans Tezi, İstanbul, 1-32 (1995).

7. Akkurt, M. , "Bilgisayar Destekli Tasarım AUTOLISP Programına Giriş", ***Birsen Yayınevi***, İstanbul, 2-15 (1998).

8. Sıngh, N., "System Approach to Computer-Integrated Design and Manufacturing", *USA*, 3-30 (1996).

9. Fellows, J. W., "All About CAD/CAM", United Kingdom, 3-20 (1997).

10. Koç, K. H., "Bilgisayar Destekli Üretim ve Orman Ürünleri Sanayinde Uygulaması", Doktora Tezi***, İstanbul Üniversitesi, Sosyal Bilimler Enstitüsü***. İstanbul, 3-31 (1993).

11. Kazan, H., "Bilgisayar Destekli Tasarım- Bilgisayar Destekli Üretim (CAD/CAM) ve Bir Uygulama" Yüksek Lisans Tezi***, Selçuk Üniversitesi, Sosyal Bilimler Enstitüsü,*** Konya, 4-32 (1993).

12. Erecek, A., "BDT/BDÜ Uygulamaları ve Bir Yazılım Sisteminin Tanıtımı",***Yüksek Lisans Tezi, Erciyes Üniversitesi, Sosyal Bilimler Enstitüsü***. Kayseri, 4-32 (1993).

13. Nalbant, M., "AutoCAD", İstanbul, 5-32 (1993).

14. Sevinç, Ü., "Ürün Geliştirmede Toplam Tasarım ve Bilgisayar Programları",Yüksek Lisans Tezi***, Gazi Üniversitesi, Fen Bilimleri Enstitüsü***. Ankara, 5-6 (1997).

15. AUTODESK BV., "AUTOLISP Programmers Reference Release 12", Switzerland, ***AUTODESK,*** (1997).

16. Mendi, F., "Tasarım ve Konstrüksiyon Ders Notları", ***Gazi Üniversitesi***, Ankara (1997).

17. Taşdelen, H. Avni., "Standart Makine Elemanlarının AutoCAD Ortamında AUTOLISP Yardımıyla Tasarımı**",** Yüksek Lisans Tezi , Ankara- (1998).

18. BİLGİN, Y., "Bilgisayar Destekli Tasarımda AUTOLISP Uygulaması", Yüksek Lisans Tezi, Kocaeli (1996).

19. Wiebe, E. N., Summey, J., "Assesment of Current Trends in Computer- Aided Design and Manufacturing in the Furniture Industry,"
http:// www. ncsu.edu/wiebe/pub.htm

20. Wiebe, E., N., Norton, J. J., Summey, J., "Organizational Assesment of Integration CAD and Product Data Management Tools in the Furniture Industry,Furniture Mnufacturing and Management Centre Technical Report", (1996-1997)
http:// www. ncsu.edu/wiebe/pub.htm.

EKLER

Ek-1 Kavelalı Birleştirme (kavela.lsp)

```
;Meydana gelebilecek hatayı tanımlamak için bir fonksiyon
;tanımlanıyor. Eğer s değişkeninin değeri  "Function cancelled"
;değilse oluşan hatayı bildiren mesajı verecektir.

MyRadios : dialog {
  key = "Title";
  label = "";//Title$ from lsp file
  spacer;
  : boxed_radio_row {
    key = "Radio1";
    label = "Kavela Çapı";
    width = 34.26;
    fixed_width = true;
    : radio_button {
      key = "A";
      label = "8 mm";
    }
    : radio_button {
      key = "B";
      label = "10 mm";
    }
  }
  : boxed_radio_column {
    key = "Radio2";
    label = "Kavela Adedi";
    width = 34.26;
    fixed_width = true;
    : radio_button {
      key = "2";
      label = "2 Adet";
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
    }
    : radio_button {
      key = "3";
      label = "3 Adet";
    }
    : radio_button {
      key = "4";
      label = "4 Adet";
    }
    : radio_button {
      key = "5";
      label = "5 Adet";
    }
    : radio_button {
      key = "6";
      label = "6 Adet";
    }
  }
  spacer;
  ok_only;
}
```

```
(defun c:MyRadios (/ Dcl_Id% Radio1$ Radio2$ Return#)
  (princ "\nMyRadios")(princ)
  ; Set Default Variables
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
 (if (not *MyRadios@);Unique global variable name to store dialog info
   (setq *MyRadios@ (list nil "" ""))
 );if
 (setq Radio1$ (nth 1 *MyRadios@)
       Radio2$ (nth 2 *MyRadios@)
 );setq
 ; Load Dialog
 (setq Dcl_Id% (load_dialog "MyDialogs.dcl"))
 (new_dialog "MyRadios" Dcl_Id%)
 ; Set Dialog Initial Settings
 (set_tile "Title" " KAVELA ")
 (set_tile "Radio1" Radio1$)
 (set_tile "Radio2" Radio2$)
 ; Dialog Actions
 (action_tile "Radio1" "(setq Radio1$ $value)")
 (action_tile "Radio2" "(setq Radio2$ $value)")
 (setq Return# (start_dialog))
 ; Unload Dialog
 (unload_dialog Dcl_Id%)
 (setq *MyRadios@ (list nil Radio1$ Radio2$))
 (princ)
);defun c:MyRadios

(defun myerror(s)
 (if(/= s"Function cancelled")
 (princ(strcat "\nError:"s))
 )


;Meydana gelebilecek hatayı tanımlamak için bir fonksiyon
;tanımlanıyor. Eğer s değişkeninin değeri  "Function cancelled"
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
;değilse oluşan hatayı bildiren mesajı verecektir.

 (defun c:kavela(/ x1 x2 sayi)
 (setvar "cmdecho" 0)
 (command "ucs" "world" "")
 (command "zoom" "all" "")

 (setq kvls(getint "\nKavela Adedini Giriniz ="))

    (if ( kvls = 1)
     (prompt "1 kavela koyulacak")

 (setq cap(getint "\nKavelanın Çapını Giriniz [8-10]="))
 (setq ycap (/ cap 2))
 (princ ycap)
 (prompt "nKAVELA DELİNECEK YÜZEYİ SEÇİNİZ")
 (setq a(getpoint"\n Yüzeyin Başlangıç Noktasını Seçiniz: "))
 (setq x2(getpoint"\n Yüzeyin Bitiş Noktasını Seçiniz:"))
 (setq sayi(distance a x2))
  )

  (if ( kvls = 2)
        (progn
     (prompt "Kavelaların hazırlanması")

        (setq kvl1 (list (+(car a)25) (cadr a) (-(caddr a)22)))
        (command "cylinder" kvl1 ycap 37 "change" "last" "" "p" "color" "7" "")

        (setq p1(getpoint "\nKavela yerleştirilecek 1. parçayı seçin:"))
        (setq nesne1(ssget p1))
        (command "subtract" nesne1 "" "last" "")
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
    (command "cylinder" kvl1 ycap 37 "change" "last" "" "p" "color" "7" "")

    (setq p2(getpoint "\n2.Parçayı seçin:"))
    (setq nesne2(ssget p2))
    (command "subtract" nesne2 "" "last" "")
    (setq k2 (list (+(car a)25) (cadr a) (-(caddr a)22)))

    ;1. kavelanın yapımı ve yerleştirilmesi
    (if (= 10 cap)
     (progn
    (command "color" "1" "")
    (command "cylinder" kvl1 ycap 35 "change" "last" "" "p" "color" "1" "")
    (setq kavelafillet_alt1 (list (+(car a)20) (cadr a) (-(caddr a)22)))
  (command "fillet" kavelafillet_alt1 "1" "")

    (setq kavelafillet_ust1 (list (+(car a)20) (cadr a) (+(caddr a)13)))
  (command "fillet" kavelafillet_ust1 "1" "")
    (setq kavelakopya (list (+(car a)100) (cadr a) (caddr a)))
     )
    )

    (if (= 8 cap)
     (progn
    (command "color" "1" "")
    (command "cylinder" kvl1 ycap 35 "change" "last" "" "p" "color" "1" "")
    (setq kavelafillet_alt1 (list (+(car a)21) (cadr a) (-(caddr a)22)))
  (command "fillet" kavelafillet_alt1 "1" "")

    (setq kavelafillet_ust1 (list (+(car a)21) (cadr a) (+(caddr a)13)))
  (command "fillet" kavelafillet_ust1 "1" "")
    (setq kavelakopya (list (+(car a)100) (cadr a) (caddr a)))
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
          )
          )

          ;girilen noktaya parçanın boyunun eklenmesi
          (setq se (list (+(car a)sayi) (cadr a) (caddr a)))

          ;2. kavelanın yerinin belirlenmesi
          (setq kvl2 (list (-(car se)25) (cadr se) (-(caddr se)22)))
       (command "cylinder" kvl2 ycap 37 "change" "last" "" "p" "color" "7" "")
       (command "subtract" nesne1 "" "last" "")

          (command "cylinder" kvl2 ycap 37 "change" "last" "" "p" "color" "7" "")
          (command "subtract" nesne2 "" "last" "")
          (command "copy" "last" "" "m" kvl1 kvl2 "")
       )

(if( kvls = 3)
   (progn
          (setq kvl_1 (list (+(car a)25) (cadr a) (-(caddr a)22)))
          (command "cylinder" kvl_1 ycap 37 "change" "last" "" "p" "color" "7" "")



          (setq p_1(getpoint "\nKavela yerleştirilecek 1. parçayı seçin:"))
          (setq nesne_1(ssget p_1))
          (command "subtract" nesne_1 "" "last" "")

          (command "cylinder" kvl_1 ycap 37 "change" "last" "" "p" "color" "7" "")

          (setq p_2(getpoint "\n2.Parçayı seçin:"))
          (setq nesne_2(ssget p_2))
          (command "subtract" nesne_2 "" "last" "")
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
    (setq k_2 (list (+(car a)25) (cadr a) (-(caddr a)22)))

    ; 1. kavelanın yapımı ve yerleştirilmesi   500mm üzeri
    (if (= 10 cap)
     (progn
    (command "color" "1" "")
    (command "cylinder" kvl_1 ycap 35 "change" "last" "" "p" "color" "1" "")
    (setq kavelafillet_alt1 (list (+(car a)20) (cadr a) (-(caddr a)22)))
  (command "fillet" kavelafillet_alt1 "1" "")

    (setq kavelafillet_ust1 (list (+(car a)20) (cadr a) (+(caddr a)13)))
  (command "fillet" kavelafillet_ust1 "1" "")
    (setq kavelakopya (list (+(car a)100) (cadr a) (caddr a)))
     )
    )

    (if (= 8 cap)
     (progn
    (command "color" "1" "")
    (command "cylinder" kvl_1 ycap 35 "change" "last" "" "p" "color" "1" "")
    (setq kavelafillet_alt1 (list (+(car a)21) (cadr a) (-(caddr a)22)))
  (command "fillet" kavelafillet_alt1 "1" "")

    (setq kavelafillet_ust1 (list (+(car a)21) (cadr a) (+(caddr a)13)))
  (command "fillet" kavelafillet_ust1 "1" "")
    (setq kavelakopya (list (+(car a)100) (cadr a) (caddr a)))
     )
    )
```

Ek-1 (Devam) Kavelalı Birleştirme (kavela.lsp)

```
        ;2. kavelanın yerinin belirlenmesi
        ;parça boyu ortasının eklenmesi

        (setq kvl_2 (list (+(car a) (/ sayi 2)) (cadr a) (-(caddr a)22)))
        (command "cylinder" kvl_2 ycap 37 "change" "last" "" "p" "color" "7" "")
      (command "subtract" nesne_1 "" "last" "")

        (command "cylinder" kvl_2 ycap 37 "change" "last" "" "p" "color" "7" "")
        (command "subtract" nesne_2 "" "last" "")
        (command "copy" "last" "" "m" kvl_1 kvl_2 "")

        ;3. kavelanın yerinin belirlenmesi
        ;girilen noktaya parçanın boyunun eklenmesi
        (setq son (list (+(car a)sayi) (cadr a) (caddr a)))

        (setq kvl_3 (list (-(car son)25) (cadr son) (-(caddr son)22)))
      (command "cylinder" kvl_3 ycap 37 "change" "last" "" "p" "color" "7" "")
      (command "subtract" nesne_1 "" "last" "")
        (command "cylinder" kvl_3 ycap 37 "change" "last" "" "p" "color" "7" "")
        (command "subtract" nesne_2 "" "last" "")
        (command "copy" "last" "" "m" kvl_2 kvl_3 "")
        )
    )
  )
)
```

Ek-2 Zıvanalı Birleştirme (zivana.lsp)

```
;Meydana gelebilecek hatayı tanımlamak için bir fonksiyon
;tanımlanıyor. Eğer s değişkeninin değeri  "Function cancelled"
;değilse oluşan hatayı bildiren mesajı verecektir.

MyEditBoxes : dialog {
 key = "Title";
 label = "";//Title$ from lsp file
 initial_focus = "Edit1";
 spacer;
 : row {//<
  fixed_width = true;
  : column {
   width = 24.76;
   fixed_width = true;
   spacer;
   : text {
    key = "Text1";
    label = "";//Text1$ from lsp file
   }   }
  : edit_box {
   key = "Edit1";//Edit1$ from lsp file
   edit_width = 9.42;
   fixed_width = true;
  }
 }//>
 : row {//<
  fixed_width = true;
  : column {
   width = 24.76;
   fixed_width = true;
```

Ek-2 (Devam) Zıvanalı Birleştirme (zivana.lsp)

```
    spacer;
    : text {
      key = "Text2";
      label = "";//Text2$ from lsp file
    }
  }
  : edit_box {
    key = "Edit2";//Edit2$ from lsp file
    edit_width = 9.42;
    fixed_width = true;
  }
 }//>
 : row {//<
  fixed_width = true;
  : column {
    width = 24.76;
    fixed_width = true;
    spacer;
    : text {
      key = "Text3";
      label = "";//Text3$ from lsp file
    }
  }
  : edit_box {
    key = "Edit3";//Edit3$ from lsp file
    edit_width = 9.42;
    fixed_width = true;
  }
 }//>
 spacer;
 ok_only;
```

Ek-2 (Devam) Zıvanalı Birleştirme (zivana.lsp)

```
}
(defun c:MyEditBoxes (/ Dcl_Id% Edit1$ Edit2$ Edit3$ Return#)
  (princ "\nMyEditBoxes")(princ)
  ; Set Default Variables
  (if (not *MyEditBoxes@);Unique global variable name to store dialog info
    (setq *MyEditBoxes@ (list nil "" "" ""))
  );if
  (setq Edit1$ (nth 1 *MyEditBoxes@)
        Edit2$ (nth 2 *MyEditBoxes@)
        Edit3$ (nth 3 *MyEditBoxes@)
  );setq
  ; Load Dialog
  (setq Dcl_Id% (load_dialog "MyDialogs.dcl"))
  (new_dialog "MyEditBoxes" Dcl_Id%)
  ; Set Dialog Initial Settings
  (set_tile "Title" " ZIVANA ")
  (set_tile "Text1" "Kayıt Kalınlığı  (mm)")
  (set_tile "Edit1" Edit1$)
  (set_tile "Text2" "Kayıt Yüksekliği (mm)")
  (set_tile "Edit2" Edit2$)
  (set_tile "Text3" "Ayak Kesiti      (mm)")
  (set_tile "Edit3" Edit3$)
  ; Dialog Actions
  (action_tile "Edit1" "(setq Edit1$ $value)")
  (action_tile "Edit2" "(setq Edit1$ $value)")
  (action_tile "Edit3" "(setq Edit1$ $value)")

  (setq Return# (start_dialog))
  ; Unload Dialog
  (unload_dialog Dcl_Id%)
  (setq *MyEditBoxes@ (list nil Edit1$ Edit2$ Edit3$))
```

Ek-2 (Devam) Zıvanalı Birleştirme (zivana.lsp)

```
  (princ)
);defun c:MyEditBoxes


(defun myerror(s)
  (if(/= s"Function cancelled")
  (princ(strcat "\nError:"s))
  )

(defun c:zıvana()
  (setvar "cmdecho" 0)
  (command "ucs" "world" "")
  (command "zoom" "a")
  (setq kg (getint "\nKayıt Kalınlığını Giriniz="))
  (setq ky (getint "\nKayıt Yüksekliğini Giriniz="))
  (setq ak (getint "\nAyak Kesitini Giriniz="))

  (setq a (getpoint "\nZıvana Yerleştirilecek Noktayı Seçiniz..."))
  (setq z1 (list (car a) (-(cadr a)(/(/ kg 3)2)) (-(caddr a)ky)))
  (setq z2 (list (-(car z1)(- ak 5)) (+(cadr z1)(/ kg 3)) (caddr z1)))
  (command "change" "1" "" "p" "color" "red")
  (command "box" z1 z2 (- ky 10) )

  (prompt "\nKayıt ve Zıvanayı seçip enter tuşuna basınız...")
  (setq ss (ssget ))
  (command "union" ss "")

  (setq ayak (getpoint "\nAyak Kesitini Seçiniz..."))
  (setq disi (list (-(car z1)(- ak 4)) (+(cadr z1)(/ kg 3)) (caddr z1)))
  (command "box" z1 disi (- ky 10) )
  (setq nesne1 (ssget ayak))
```

Ek-2 (Devam) Zıvanalı Birleştirme (zivana.lsp)

```
  (command "subtract" nesne1 "" "last" "")
  (setq kfillet1 (list (-(car a)5) (-(cadr a)(/(/ kg 3)2)) (-(caddr a)10)))
  (setvar "filletrad" (/ kg 6))

  (prompt "Pahlandırılacak kenarları seçiniz...")
  (command "fillet" kfillet1 "")
  )
)
```

ÖZGEÇMİŞ

Kişisel Bilgiler

Soyadı, adı	: BULUT, Hayri
Uyruğu	: T.C.
Doğum tarihi ve yeri	: 05.11.1983 Ankara
Medeni hali	: Bekar
Telefon	: 0532 225 01 39
e-mail	: hayribulut06@hotmail.com

Eğitim

Derece	Eğitim Birimi	Mezuniyet tarihi
Yüksek Lisans	Gazi Üniversitesi/Fen Bilimleri Enst.	……
Lisans	Gazi Üniversitesi/ Mob. ve Dek. Eğt	2007
Lise	Yenimahalle Endüstri Meslek Lisesi	1999

Yabancı Dil

İngilizce

Hobiler

Bilgisayar Teknolojileri, Proje, Tasarım, Uygulama, Fuar ve Organizasyon, Seyahat,