**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ**

# SEMBOLİK PLANLAMA İÇİN UZAM-ZAMANSAL ÇIKARSAMAYLA NESNE MODELLERİNİN VE ETKİLEŞİMLERİNİN ÖĞRENİLMESİ

**YÜKSEK LİSANS TEZİ**

**Mustafa ERSEN**

**Bilgisayar Bilimleri Anabilim Dalı**

**Bilgisayar Bilimleri Programı**

**HAZİRAN 2012**

**İSTANBUL TEKNİK ÜNİVERSİTESİ ★ BİLİŞİM ENSTİTÜSÜ**

# SEMBOLİK PLANLAMA İÇİN UZAM-ZAMANSAL ÇIKARSAMAYLA NESNE MODELLERİNİN VE ETKİLEŞİMLERİNİN ÖĞRENİLMESİ

**YÜKSEK LİSANS TEZİ**

**Mustafa ERSEN**
**(704091024)**

**Bilgisayar Bilimleri Anabilim Dalı**

**Bilgisayar Bilimleri Programı**

**Tez Danışmanı: Yrd. Doç. Dr. Sanem SARIEL TALAY**

**HAZİRAN 2012**

İTÜ, Bilişim Enstitüsü'nün **704091024** numaralı Yüksek Lisans Öğrencisi **Mustafa ERSEN**, ilgili yönetmeliklerin belirlediği gerekli tüm şartları yerine getirdikten sonra hazırladığı "**SEMBOLİK PLANLAMA İÇİN UZAM-ZAMANSAL ÇIKARSAMAYLA NESNE MODELLERİNİN VE ETKİLEŞİMLERİNİN ÖĞRENİLMESİ**" başlıklı tezini aşağıda imzaları olan jüri önünde başarı ile sunmuştur.

| | | |
|---|---|---|
| **Tez Danışmanı :** | **Yrd. Doç. Dr. Sanem SARIEL TALAY**<br>**İstanbul Teknik Üniversitesi** | ..... |
| **Jüri Üyeleri :** | **Doç. Dr. Şima ETANER UYAR**<br>**İstanbul Teknik Üniversitesi** | ..... |
| | **Prof. Dr. Coşkun SÖNMEZ**<br>**Yıldız Teknik Üniversitesi** | ..... |

**Teslim Tarihi :** **03 Mayıs 2012**
**Savunma Tarihi :** **07 Haziran 2012**

***Anne ve babama,***

## ÖNSÖZ

Yüksek lisans eğitimim boyunca pek çok konuda destek aldığım, fikir ve tecrübelerinden faydalandığım ve birlikte çalışmaktan mutluluk duyduğum danışmanım Sanem SARIEL TALAY'a ve bu günlere gelmemde büyük pay sahibi olan aileme ve tüm dostlarıma teşekkürlerimi sunarım.

Haziran 2012

Mustafa Ersen
(Bilgisayar Mühendisi)

**İÇİNDEKİLER**

**Sayfa**

## KISALTMALAR

**ADL** : Action Description Language
**CNF** : Conjunctive Normal Form (Birletimli Normal Form)
**KB** : Knowledge Base (Bilgi Tabanı)
**LOCM** : Learning Object-Centred Models (Nesne-Merkezli Modellerin Öğrenilmesi)
**MBR** : Minimum Bounding Rectangle (Kapsayan En Küçük Dikdörtgen)
**PDDL** : Planning Domain Definition Language
**RCC** : Region Connection Calculus (Bölge Bağlantı Analizi)
**SDM** : Sonlu Durum Makinesi
**STRIPS** : Stanford Research Institute Problem Solver
**TIM** : The Incredible Machine bilgisayar oyunu

## ÇİZELGE LİSTESİ

**Sayfa**

## ŞEKİL LİSTESİ

**Sayfa**

## SEMBOLİK PLANLAMA İÇİN UZAM-ZAMANSAL ÇIKARSAMAYLA NESNE MODELLERİNİN VE ETKİLEŞİMLERİNİN ÖĞRENİLMESİ

## ÖZET

Yapay Zeka çalışmalarındaki ana amaçlardan biri verilen görevleri yerine getirecek ve insanların hayatını kolaylaştıracak otonom etmenlerin tasarlanması ve gerçeklenmesidir. Bir etmenin verilen bir görevi gerçekleştirebilmesi için öncelikle eylemlerinin ortamdaki etkilerini gözeterek plan yapması gerekir. Planlama aşamasında etmen, sahip olduğu ortam modeli ve problem tanımına ilişkin bilgilerden yararlanır. Yapılan planın hedefle tutarlı sonuçlar üretebilmesi problemin biçimsel tanımının ne kadar doğru yapıldığına ve plan yapan etmenin sahip olduğu biçimsel ortam modelinin ne ölçüde ortamın gerçek yapısı ile örtüştüğüne bağlıdır.

Bazı ortamlarda etmenlerin verilen probleme çözüm üretebilmesi için çeşitli nesne, alet ve makineler kullanması gerekebilir. Bu tür durumlarda başarıyla plan üretilebilmesi için, ortamdaki nesne, alet ve makinelerin işlevleri ve birbirleri ile olan ilişkileri etmen tarafından bilinmelidir. Ortamın temsiline ilişkin bu bilgiler etmene sağlanmadığında ortam modelinin gözlemlerle öğrenilmesi gerekir.

Bu çalışmada, nesneler arası etkileşimlerin ortamda gözlemlenen olaylar arasındaki durum bilgisini gerektirmeden, eğitim senaryoları kullanılarak öğrenilmesine ilişkin bir yöntem sunulmaktadır. Bu amaca uygun bir sınama ortamı olarak "The Incredible Machine" bilgisayar oyunu kullanılmıştır. Bu oyundaki senaryolarda verilen hedeflere ulaşmak üzere, oyuncunun farklı işlevlere sahip nesneleri birbirleri ile zincirleme etkileşimler içinde kullanması gerekmektedir. İlk durum ve hedef duruma sahip senaryolar üzerinden oynanan bu oyun planlama problem tanımına uygun bir örnek teşkil etmektedir.

Çalışma kapsamında, ortamdaki nesnelerin davranış ve işlevlerini modellemek üzere sonlu durum makineleri kullanılmıştır. Ayrıca, farklı nesne gruplarını modellemekte kullanılan sonlu durum makineleri arasındaki koşullu bağlantıların verilen eğitim senaryolarında gözlemlenen olaylardan faydalanılarak özerk bir şekilde oluşturulması sağlanmıştır. Olaylar üzerinden etkileşimlerin modellenmesi aşamasında nesneler arası ilişki ve bağlantıları modelleyen farklı seviyede bilgilerden yararlanılmış ve sonuçlar karşılaştırılmıştır. Etmene, nesneler arası doğrudan gözlemlenebilen ilişkilerden oluşan insan seviyesinde bir bilgi tabanı sağlandığında, planlama için kullanılacak etkileşimler istenilen seviyede öğrenilebilmektedir. Ayrıca, ortamdan otonom olarak elde edilmesi mümkün olan nesnelerin uzamsal bilgileri ile olayların zamansal bilgileri de öğrenme için kullanılabilmekte ve uzam-zamansal bir sistem altında bu iki yaklaşımın birleştirilmesi bilgi tabanlı yaklaşıma yakın sonuçlar üretmektedir. Uzam-zamansal çıkarsamanın büyük miktarda bilgi gerektirmemesi ve bilgi tabanlı yaklaşıma yakın bir sonuç üretmesi makine seviyesinde öğrenmenin başarılı olduğu sonucunu doğurmaktadır.

## LEARNING OBJECT MODELS AND INTERACTIONS THROUGH SPATIO-TEMPORAL REASONING FOR SYMBOLIC PLANNING

## SUMMARY

One of the major aims in Artificial Intelligence research is to design and implement autonomous agents in order to accomplish some useful tasks and make daily life of human beings easier. To be able to perform a given task successfully, an agent needs to devise a plan consistent with the task beforehand, using its knowledge about the preconditions and the effects of its applicable actions in the environment and the given model of the problem. The process of reasoning in order to achieve the given task is called as automated action planning. In automated action planning, agents use the model of the environment they possess and the given information on the problem. The performance of the planning process strongly depends on the correctness and the completeness of the abstract model of the environment and the problem description. Whenever the planning problem (i.e., initial and goal states) and operators corresponding to domain actions are defined properly, a planner can devise a solution to reach the goal state using various possible planning techniques. However, the problem gets complicated if the planning agent has incomplete or incorrect information about the preconditions and the effects of the domain operators. In this case, the problem turns into a learning task of action schema where the relations among actions and the environment are modelled through observations.

In some applications, the agent may need to use objects, tools and machines in the environment to devise a plan consistent with its task. A good example of this situation of the necessity of using objects in the environment can be given as the Monkey-Banana problem where a monkey is in a room containing a box, a knife and a bunch of bananas. The aim of the monkey is to get the bananas which are hanging from the ceiling out of his reach. The monkey needs to push the box under the bananas and climb on top of it in order to reach the bananas. Moreover, the monkey needs to use the knife to cut the rope connecting the bananas to the hanger. These types of problems require the agent to possess correct and complete information on the functions of objects, tools and machines and the relationships among them. When this information is not supplied, it is essential for the agent to observe the environment and gather the required knowledge to build a consistent model to be able to successfully use the objects in the environment during planning.

The main goal of this research is to develop a method for learning functions of objects and interactions among them through observation of a given sequence of events on objects without considering the intermediate state information between any pair of sequential events. We analyse the performance of the learner against the level of completeness of the knowledge base about relations among objects in the environment. We believe that the outcomes of this study are also useful for robot learning tasks. Our future work includes implementation of this system on real robots for learning affordances.

We selected the Incredible Machine computer game as a suitable domain for analysing interactions among different types of objects, tools and machines as this domain allows the use of various objects and their interactions to build complex systems in order to solve given puzzles. In a typical scenario of this game, the player is given a limited number of objects, tools and machines and a goal to accomplish by using these resources. Initially, some objects, tools and machines are given as already placed in the environment and the player needs to use this partial structure in the solution. To complete the given structure to reach the goal state, the player is provided with some objects, tools and machines in a parts bin and these parts must be used without changing the position of already existing objects in the environment and the connections among them. The puzzles in the game are interesting as a research domain for our study because they require building complex systems by considering various interactions among different objects to achieve the given goals.

This problem fits into the planning framework as it involves events to transform a given initial state into a goal state. Both progression and regression planning is applicable if domain events are perfectly defined. A planner needs to consider the game in two aspects to be able to devise a consistent plan. First, considering interactions among objects, tools and machines is crucial for making decisions while connecting given parts altogether to initiate chain reactions to reach the goal state. Second, physics models of some objects in the domain (e.g., the effect of the gravity on the motion of a ball) should be taken into account to estimate interactions through motions.

In this work, we focus on solving the stated learning problem where preconditions and effects of events on domain objects are not known but interactions of objects are given in a tutorial to understand the rationale behind the Incredible Machine puzzles. We mainly concentrate on the automated modelling of behaviours of objects, tools and machines in the domain and learning the interactions among them through events. During learning, our system mostly uses qualitative approaches and we omit the quantitative part of exact reasoning on the physics model and motion of some objects such as the balls. We assume that a given tutorial presents object interactions in the form of chain reactions (e.g., starting a mixer, running a motor, lighting a lamp, etc.) in a text-based format. No further background information is available about the types of objects and their semantics, nor intermediate state information.

In our system, objects, tools and machines are modelled by using state machines for each different type of objects to capture the behaviour of them. Automated learning of interactions among various objects through observation of events is accomplished by adding conditional links among state machines in order to model preconditions and conditional effects. In this part of determining conditional links, we investigate the problem from four perspectives and propose an integrated solution. If directly observable relations of parts are provided to the agent, a logic-based reasoning mechanism could be used to learn interactions through observation of events. Otherwise, we show that considering spatial, temporal and spatio-temporal aspects of the problem makes it possible to reason about interactions for further planning tasks. Our system uses mereotopological and directional information of objects and temporal information of events to model relations among objects in order to learn interactions when a knowledge base of relations is not supplied. These four perspectives may correspond to different learning problems. The Incredible Machine domain provides a suitable framework to integrate these approaches.

In our experiments using various tutorials, we show that learning is accomplished to a desired extent when a human-level knowledge base of directly observable relations among objects, tools and machines is provided to the agent. Our analysis also indicates that using spatial information of objects and temporal information of events makes it feasible to learn interactions when a knowledge base is not available beforehand. Moreover, combining spatial and temporal data in a spatio-temporal approach is superior to using spatial and temporal approaches alone and gives close results to that of the knowledge-based approach. This is promising because using spatio-temporal information does not require great amount of knowledge and this information can easily be extracted from the domain without human-level intervention.

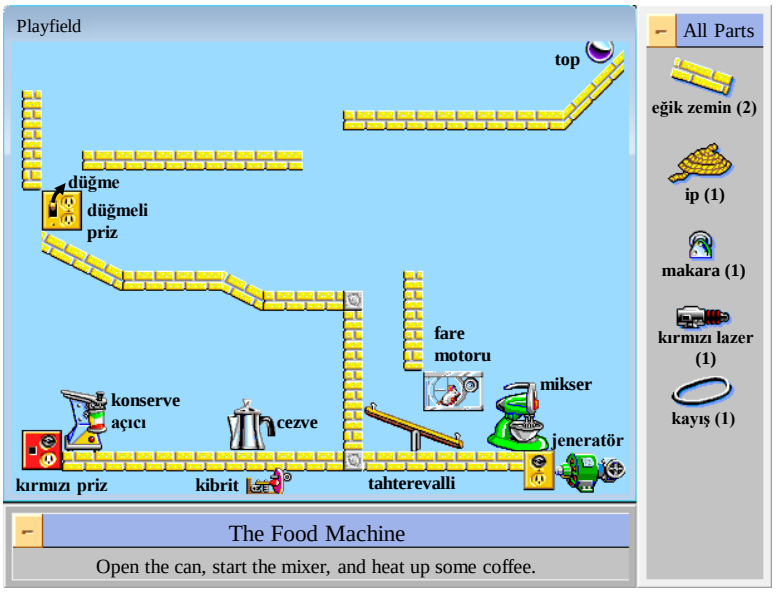
# 1. GİRİŞ

Özerk eylem planlama, belirli bir başlangıç durumundan belirlenen bir hedefe ulaşmak için yürütülmesi gereken eylemler dizisinin oluşturulması sürecidir. Ortamın ilk durumu ve hedef durumu ile ortamda etmen tarafından gerçekleştirilebilen eylemler uygun şekilde (önkoşul ve etkileri ile birlikte) verildiğinde, planlama sistemleri kullanılarak çözüme yönelik bir plan ileri yönde (*progression*) ya da geri yönde (*regression*) aramaya dayalı olarak geliştirilebilir. Bununla birlikte, ortam ve eylemler hakkında yeterli bilgiye sahip olunmadığında sembolik planlama yapılabilmesi için gerekli bileşenlerin ortamdan gözlemlerle öğrenilmesi gerekir.

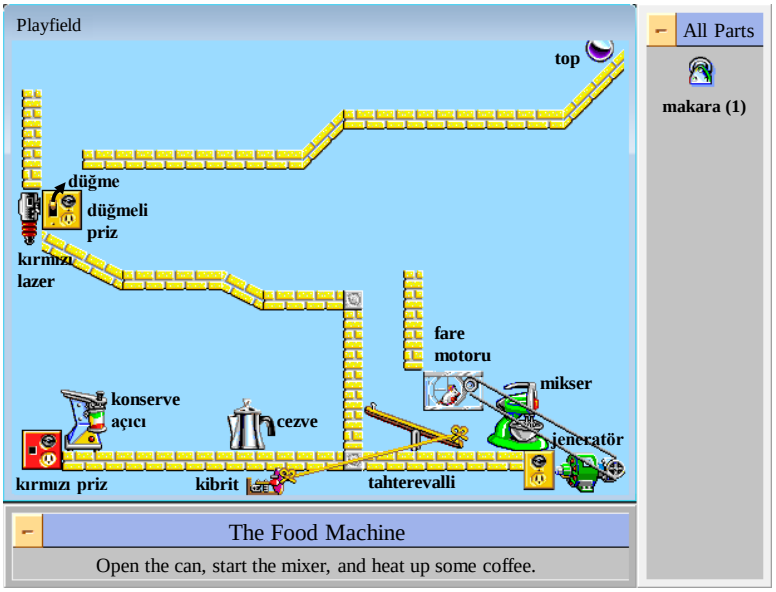
Bazı ortamlarda etmenlerin verilen probleme çözüm üretebilmesi için ortamdaki çeşitli nesne, alet ve makinelerden yararlanması gerekebilir. Bu tür durumlarda başarıyla plan üretilebilmesi için, ortamdaki nesnelerin işlevlerini ve birbirleri ile olan ilişkilerini modelleyen sembolik eylem temsilleri etmen tarafından bilinmelidir. Ortam temsiline ilişkin bu bilgiler etmen tarafından bilinmediğinde bu bilgilerin ortamdan gözlemlerle etmen tarafından otonom olarak elde edilmesi gerekir.

The Incredible Machine (TIM) adlı bilgisayar oyunu bu duruma güzel bir örnek teşkil etmektedir (The Incredible Machine 3 [PC Software], 1995). Bu oyunda verilen hedeflere ulaşmak için, oyuncunun verilen kısıtlı sayı ve türdeki nesneleri birbirleri ile zincirleme etkileşim içinde kullanması gerekmektedir. Bunun için öncelikle ortamdaki nesnelerin birbirleri üzerindeki etkilerinin ve etkileşim koşullarının bilinmesi gerekir. Oyunda verilen problemlerin çözümüne yönelik olarak oluşturulan sistemler Rube Goldberg'in makinelerine benzemektedir (Rube Goldberg Inc., 2012).

Şekil 1.1 (a)'da örnek bir TIM probleminin başlangıç durumu verilmiştir. Bu örnek problemde hedef, konservenin açılması, mikserin çalışması ve kahvenin hazırlanmasıdır. Hedef duruma ulaşmak için kullanılmak üzere, şeklin sağındaki parça kutusunda sınırlı çeşit ve sayıda nesneler verilmiştir. Bu nesnelerin ortamda bulunan diğer bileşenlerin konum ve durumları değiştirilmeden kullanılması gerekmektedir.




**(a)**




**(b)**

**Şekil 1.1:** Örnek bir TIM probleminin (a) başlangıç ve (b) hedef durumları.

Ortamdaki nesnelerden konserve açıcı, mikser ve kırmızı lazer elektrikle çalışan nesnelerdir. Konserve açıcı, kırmızı lazerin temasıyla elektrik veren kırmızı prize takılı durumdadır ve mikser ise jeneratöre bağlanmıştır. Düğmeli priz, düğmesi aşağı itildiğinde kendisine bağlanan cihazlara elektrik sağlayabilmektedir. Top ortamda yer çekiminin etkisiyle hareket etmektedir. Fare motoru herhangi bir nesnenin çarpması ile harekete geçerek kendisine kayış ile bağlanan bir takım sistemleri döndürebilmektedir. Cezvenin altındaki kibrit bir ip yardımıyla çekildiğinde yanmaktadır. Tahterevalli, top vb. bir nesnenin çarpmasıyla veya ucundan iple çekilerek hareket ettirilmekte ve ucuna bağlı durumdaki ipi çekebilmektedir.

Verilen nesneler Şekil 1.1 (b)'deki gibi konumlandırıldığında ve gerekli bağlantılar (ip ve kayış) kurulduğunda problemin çözümüne ulaşılır. Çözümü sağlayan bu durumda, eğik zeminden aşağı hareket eden top düğmeli prizin düğmesini aşağı iterek bu prize bağlı olan kırmızı lazerin çalışmasını sağlar. Kırmızı lazer aşağıdaki kırmızı prize ulaştığında bu priz de elektrik vermeye başlar ve buna bağlı olan konserve açıcı çalışır. Bu esnada eğik zeminin etkisiyle yoluna devam eden top tahterevallinin üzerine düşerek ucuna bağlı olan ipin çekilmesini ve kibritin yanmasını sağlar. Kibritin yanmasıyla birlikte cezvedeki kahve ısınır. Ayrıca, tahterevallinin ucu fare motoruna çarparak farenin koşmasını ve oluşan döndürme etkisi ile fare motoruna bağlı olan jeneratörün çalışmasını sağlar. Jeneratör çalışınca ondan elektrik alan mikser de çalışır ve böylece hedeflenen işlemler gerçekleşir.

Örnek problemde görüldüğü gibi, bu oyunda verilen problemlere çözüm üretilirken problemin iki farklı açıdan değerlendirilmesi gerekmektedir. Öncelikle, ortamdaki nesnelerin işlevlerinin ve birbirleri ile hangi koşullarda ve ne şekilde etkileşim kurabildiklerinin bilinmesi gerekmektedir. Bunun yanı sıra, top gibi hareket halindeki nesnelerin fizik modelleri ile yer çekiminin harekete etkisi de hesaba katılmalıdır. Tez çalışması kapsamında, problemin ana kısmı olan nesne işlev ve etkileşimlerinin belirlenmesi üzerine çalışılmış ve ortamın fizik modeli ihmal edilmiştir.

## 1.1 Tezin Amacı ve Katkılar

Bu çalışmanın amacı, TIM ortamında yeni problemlerin çözümünde kullanılmak üzere, verilen eğitim senaryolarında gözlemlenen olaylardan yola çıkarak nesnelerin işlevleri ve birbirleri ile etkileşimlerini öğrenen bir sistem sunmaktır. Bu amaçla çalışma kapsamında, oyundaki çeşitli nesneler üzerinde gerçekleşen olaylara ilişkin metin-tabanlı eğitim senaryoları oluşturulmuştur. Bu senaryolara ek olarak nesneler arası ilişkilere ait bilginin gösteriminde problemi farklı açılardan değerlendiren bir yaklaşım geliştirilerek farklı çıkarsama mekanizmaları incelenmiştir. Öncelikle, nesneler arasında doğrudan gözlemlenebilen bağlantıları kullanan bilgi tabanlı bir çıkarsama mekanizması ile nesnelerin birbirleri üzerindeki koşullu etkilerini belirleyen bir sistem oluşturulmuştur. Daha sonra, bilgi tabanı oluşturmanın zorluğu göz önüne alınarak ortamdan kolaylıkla elde edilebilen nesnelere ilişkin uzamsal

bilgiler ve olaylara ilişkin zamansal bilgilerden çıkarım yapılmasına dayalı yöntemler geliştirilmiş ve tüm yöntemler için deney sonuçları analiz edilmiştir.

Bu tez çalışmasının katkıları aşağıdaki şekilde sıralanabilir:

- Bir ortamda bulunan nesneler arası etkileşimlerin gözlem yoluyla öğrenilerek planlama yapılabilmesi için bir öğrenme sisteminin sunulması.
- Öğrenme ve planlama yöntemlerini bir arada kullanılabilmek üzere bir bilgisayar oyunun planlama problemi şeklinde modellenmesi ve literatüre yeni bir planlama/öğrenme domeninin kazandırılması.
- Nesne davranışlarının modellenmesi için literatürde mevcut bulunan bir yönteme farklı nesne yönelimlerine göre nesneye ilişkin olaylardaki etki değişimini göz önüne alacak şekilde eklenti yapılması.
- Ortamdaki nesneler arası koşullu etkileşimi modellemek için olaylara ilişkin zamansal bilgi ve nesnelere ilişkin uzamsal bilgiden faydalanılmasının sağlanması.
- Robotlarda planlamada nesne kullanımına yönelik sembolik planlama operatörlerinin ortamdan gözlemlerle öğrenilmesi için zemin hazırlanması.

Tezin ileriki bölümlerinin organizasyonu şu şekildedir: 2. bölümde bilgi temsili ve mantıksal çıkarsama konusunda temel bilgi verildikten sonra, 3. bölümde uzamsal ve zamansal bilgilerin çıkarsamada kullanımına ilişkin önceki çalışmalardan bahsedilmektedir. 4. bölümde, sembolik planlama konusunda temel bilgi verilmekte ve plan operatörlerinin öğrenilmesi konusundaki çeşitli yaklaşımlar ele alınmaktadır. 5. bölümde problem tanımı verilerek çözüm için önerilen sistem genel olarak tanıtıldıktan sonra, 6 ve 7. bölümlerde iki aşama halinde çalışan sistemin detayları anlatılmaktadır. İlk aşamada nesnelerin davranışları sonlu durum makineleri ile modellenmekte ve ikinci aşamada nesneler arası etkileşimlerin öğrenilmesinde nesneler arası ilişkiyi modelleyen dört farklı seviyede bilgi kullanımına dayalı bir yaklaşım sunulmaktadır. 7. bölümün sonunda öğrenilen etkileşimlerin örnek bir problemin çözümüne yönelik sembolik planlama esnasında kullanımı gösterilmekte ve 8. bölümde öğrenmede kullanılan yaklaşımlar deneylerle sınanmaktadır. Son olarak 9. bölümde sonuçlar tartışılmakta ve ileri çalışmalardan bahsedilmektedir.

# 2. BİLGİ TEMSİLİ VE MANTIKSAL ÇIKARSAMA

Otonom bir etmenin verilen görevleri yerine getirebilmesi için bulunduğu ortamı tanıması ve ortamda yürütebileceği eylemlerin önkoşul ve etkilerinden haberdar olması gerekir. Bu nedenle etmenler tasarlanırken, bilginin ne şekilde temsil edileceği ve sahip olunan bilgi üzerinde ne tür mantıksal çıkarsama mekanizmalarının kullanılacağı belirlenir. Sembolik bilgi temsilinde önermeler mantığı (*propositional logic*) veya yüklem mantığı (*first-order/predicate logic*) kullanılarak bir bilgi tabanı (*knowledge base – KB*) oluşturulur. Oluşturulan bu bilgi tabanı üzerinden mantıksal çıkarsama yapılırken çözümleme (*resolution*), ileri yönde çıkarsamalar zinciri (*forward chaining*) ve geri yönde çıkarsamalar zincirine (*backward chaining*) dayalı yöntemlerden yararlanılır.

## 2.1 Etmenlerde Bilginin Temsil Edilmesi

Bilgi tabanlı etmenlerde tasarım aşamasında kodlanan ön bilgi ile ortamdan gözlemlerle öğrenilen bilginin mantıksal temsili önermeler mantığı (*propositional logic*) veya yüklem mantığı (*first-order/predicate logic*) kullanılarak yapılabilir.

### 2.1.1 Önermeler mantığı

Şekil 2.1'de görülebileceği gibi önermeler mantığında bilgi temsil edilirken yalın ifadeler (önerme) ve bu ifadeleri birleştiren bağlaçlardan yararlanılır. Yalın ifadeler, doğru ya da yanlış şeklinde değer alabilen tek bir sembolden oluşurlar. Birleşik ifadeler ise birden fazla ifadenin mantıksal bağlaçlarla ($\wedge$, $\vee$, $\Rightarrow$, $\Leftrightarrow$ vs.) bağlanmasıyla elde edilirler. Bir ifadenin değili de (tümlenmiş hali, $\neg$) birleşik ifade olarak kabul edilir.

Önermeler mantığı ile bilgi temsilinde bir takım kısıtlamalar bulunmaktadır. Öncelikle, bu temsille varlıkların özellikleri ve varlıklar arasındaki ilişkiler doğrudan ifade edilemez. Ayrıca sadece önermeler kullanılarak genellemelerin (örneğin, bütün kuşlar kanatlıdır) ifade edilmesi zordur.

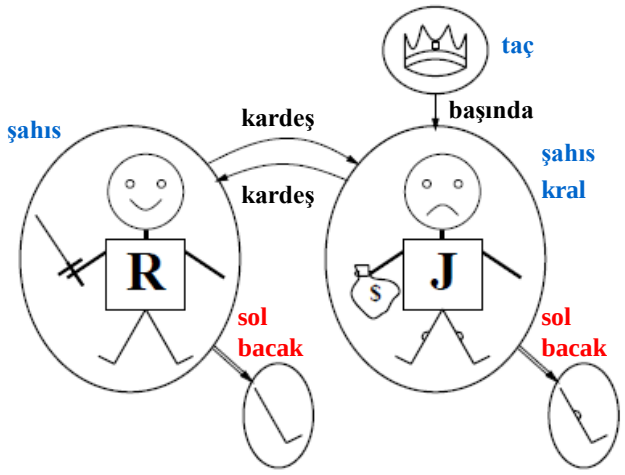
$$
\begin{aligned}
&ifade \rightarrow yalın\ ifade \mid birleşik\ ifade \\
&yalın\ ifade \rightarrow sembol \mid \textbf{doğru} \mid \textbf{yanlış} \\
&sembol \rightarrow \textbf{P} \mid \textbf{Q} \mid \textbf{R} \mid \ldots \\
&birleşik\ ifade \rightarrow \neg\, ifade \\
&\qquad\qquad \mid (ifade \wedge ifade) \\
&\qquad\qquad \mid (ifade \vee ifade) \\
&\qquad\qquad \mid (ifade \Rightarrow ifade) \\
&\qquad\qquad \mid (ifade \Leftrightarrow ifade)
\end{aligned}
$$

**Şekil 2.1:** Önermeler mantığındaki ifade yapısı (Russell & Norvig, 2002).

Önermeler mantığındaki birleşik ifadelere örnek olarak P = "Bütün insanlar ölümlüdür.", Q = "Socrates bir insandır." ve R = "Socrates ölümlüdür." yalın önermelerinin uygun şekilde bir araya getirilmesiyle elde edilen $P \wedge Q \Rightarrow R$ ifadesi verilebilir.

### 2.1.2 Yüklem mantığı

Yüklem mantığıyla bilgi temsilinde, varlıkların özellikleri ve varlıklar arasındaki ilişkiler birinci dereceden mantıksal ifadeler ve fonksiyonlar yardımıyla gösterilir. Ayrıca, niceleyiciler (*quantifiers:* ∀ ve ∃) kullanılarak genellemelerin kolayca ifade edilmesine olanak tanınır. Şekil 2.2'de yüklem mantığı ile bilgi temsiline ilişkin bir örnek verilmiştir. Bu örnekte, yuvarlak içinde verilen 5 varlık arasındaki ilişkiler 2 adet ikili ilişki (*kardeş* ve *başında*), 3 adet tekli ilişki (*şahıs*, *kral* ve *taç*) ve 1 adet fonksiyon (*sol bacak*) ile modellenmektedir.




**Şekil 2.2:** Yüklem mantığı ile bilgi temsiline örnek (Russell & Norvig, 2002).

## 2.2 Mantıksal Çıkarsama Yöntemleri

Otonom etmenler, ortamda verilen görevleri yerine getirmek için, önerme mantığı veya yüklem mantığı kullanılarak oluşturulan bilgi tabanı ($KB$) üzerinden gerektirmelere (*entailment*) dayalı mantıksal çıkarsama yaparlar. Bilgi tabanının bir α ifadesini gerektirmesi ($KB \models \alpha$) için $\alpha$'nın $KB$'nin sağlandığı her durumda doğru olması gerekir. Örneğin, bilgi tabanında P = "Bütün insanlar ölümlüdür." ifadesini barındıran bir etmen, Q = "Socrates bir insandır." gözleminden sonra R = "Socrates ölümlüdür." sonucunu çıkarabilir ($P \wedge Q \models R$). Yaygın olarak kullanılan üç temel çıkarsama yöntemi bulunmaktadır: çözümleme (*resolution*), ileri yönde çıkarsamalar zinciri (*forward chaining*) ve geri yönde çıkarsamalar zinciri (*backward chaining*).

### 2.2.1 Çözümleme

Çözümleme ile çıkarsama, tam ve sonuca ulaşan (*complete*) bir çıkarsama yöntemidir. $KB$'den çıkarılabilecek tüm $\alpha$ ifadelerini bulur. Bu yaklaşımda bir $\alpha$ ifadesinin $KB$ bilgi tabanından elde edilebildiğini ($KB \models \alpha$) göstermek için olmayana ergiden (*proof by contradiction*) yararlanılarak $KB \wedge \neg\alpha$ ifadesinin geçersiz olduğu ispatlanmaya çalışılır. $KB \wedge \neg\alpha$ üzerinde çözümleme yapılabilmesi için, öncelikle bu ifadelerin birletimli normal forma (*conjunctive normal form – CNF*) dönüştürülmeleri gerekmektedir. Bu formdaki birleşik önermeler mantıksal $VEYA$ işlemiyle (*disjunction*) birleştirilmiş ifadelerin, mantıksal $VE$ işlemi (*conjunction*) ile bir araya getirilmesiyle temsil edilir. Çözümleme algoritmasında birletimli normal forma dönüştürülmüş ifadeler üzerinde yanlış (*false*) önermesine ulaşana dek birim çözümleme yöntemi uygulanır. Birim çözümleme işlemi $(A \vee B) \wedge \neg B \Rightarrow A$ şeklinde yapılır. Birim çözümlemeler ile olası ifadeler üzerinde tüm kombinasyonlar elde edilir. Yanlış önermesine ulaşıldıysa, $KB \wedge \neg\alpha$ ifadesinin yanlış olduğu gösterilmiş olarak $KB \models \alpha$ çıkarımının mümkün olduğu sonucuna varılır. Aksi durumda $KB \models \alpha$ çıkarımının yapılamayacağı gösterilmiş olur.

Çözümleme ile çıkarsamaya örnek olarak, bilgi tabanında "köpekler kemik sever", "köpekler sevdikleri her şeyi yer" ve "Çakıl bir köpektir" ifadelerine sahip olan bir etmenin "Çakıl kemik yer" sonucuna ulaşmaya çalıştığını varsayalım.

$\boldsymbol{KB}$: $\big(köpek(x) \Longrightarrow sever(x, kemik)\big) \wedge \big(köpek(x) \wedge sever(x, y) \Longrightarrow yer(x, y)\big)$
$\wedge\, köpek(Çakıl)$

$\boldsymbol{\alpha}$: $yer(Çakıl, kemik)$

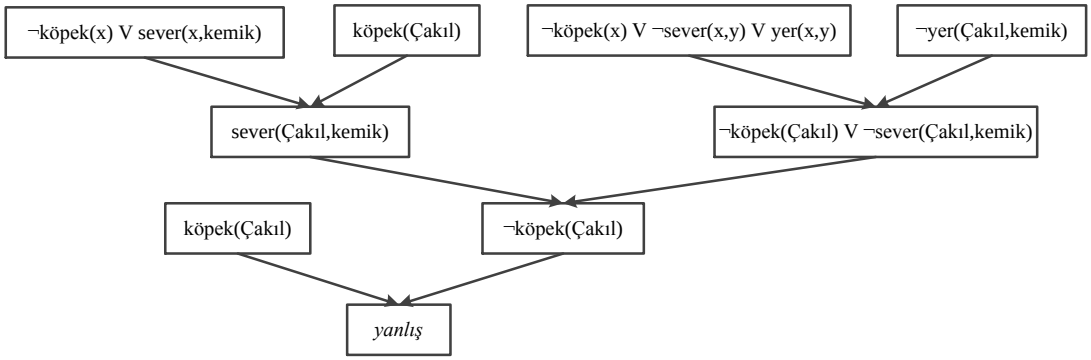
Çözümleme işlemi, $KB \wedge \neg\alpha$ ifadesinin sağlanamadığının gösterilmesi şeklinde yapıldığından önce bu ifade oluşturulur.

$$\boldsymbol{KB \wedge \neg\alpha}: \big(köpek(x) \Longrightarrow sever(x, kemik)\big)$$
$$\wedge \big(köpek(x) \wedge sever(x, y) \Longrightarrow yer(x, y)\big)$$
$$\wedge köpek(Çakıl)$$
$$\wedge \neg yer(Çakıl, kemik)$$

Çözümleme öncesinde, $KB \wedge \neg\alpha$ ifadesi birletimli normal forma dönüştürülür.

$$\boldsymbol{KB \wedge \neg\alpha}: \big(\neg köpek(x) \vee sever(x, kemik)\big)$$
$$\wedge \big(\neg köpek(x) \vee \neg sever(x, y) \vee yer(x, y)\big)$$
$$\wedge köpek(Çakıl)$$
$$\wedge \neg yer(Çakıl, kemik)$$

Birletimli normal formdaki $KB \wedge \neg\alpha$ üzerinde çözümleme Şekil 2.3'teki gibi yapılır. Birim çözümleme işlemi $(A \vee B) \wedge \neg B \Rightarrow A$ şeklinde yapılır ve uygun durumlarda değişken değeri yerine konulur (örneğin, $köpek(x) \Rightarrow köpek(Çakıl)$).



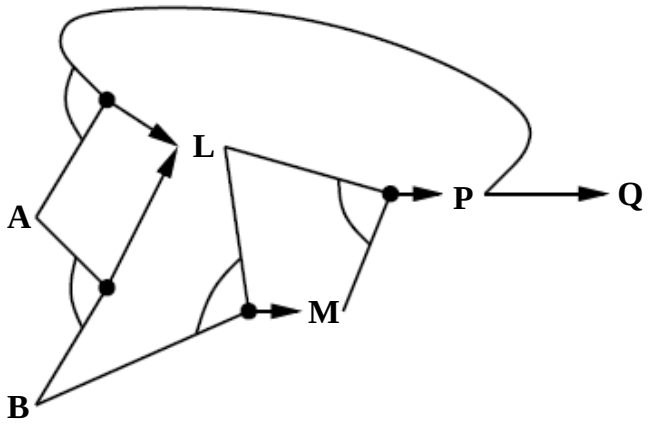

**Şekil 2.3:** Çözümleme yöntemiyle çıkarsama örneği.

Gerçek dünyadaki bilgi tabanları kısıtlı bir temsil olan Horn formunda (Horn, 1951) ifade etmeye uygun olduğundan, tüm ifade kombinasyonlarının denendiği çözümlemenin yanı sıra doğrusal karmaşıklıktaki çıkarsama mekanizmaları olan ileri yönde ve geri yönde çıkarsamalar zincirleri ile de çıkarım yapılabilmektedir. Horn formundaki önermeler en çok biri pozitif olan yalın ifadelerin mantıksal ayırtımı (*disjunction*) şeklinde olup, öncülleri pozitif ifadelerin birletimi (*conjunction*) ve sonucu tek bir pozitif ifade olan koşullu önermelere dönüştürülebilirler.

### 2.2.2 İleri yönde çıkarsamalar zinciri

İleri yönde çıkarsamalar zincirinde, her adımda bilgi tabanında öncülleri (*premise*) sağlanan ifadelerin sonuçları bilgi tabanına eklenerek ispatlanmak istenen önermeye ulaşılmaya çalışılır. Şekil 2.4'te karşı düşen VE-VEYA çizgesi (*AND-OR Graph*) ile birlikte verilen bilgi tabanı üzerinde ileri yönde çıkarsama yapılarak $KB \vDash Q$ sonucuna 4 adımda varılabilir. İşlem adımları $A \wedge B \Longrightarrow L$, $B \wedge L \Longrightarrow M$, $L \wedge M \Longrightarrow P$ ve $P \Longrightarrow Q$ şeklindedir.

$P \Longrightarrow Q$
$L \wedge M \Longrightarrow P$
$B \wedge L \Longrightarrow M$
$A \wedge P \Longrightarrow L$
$A \wedge B \Longrightarrow L$
$A$
$B$




**Şekil 2.4:** Basit bir bilgi tabanı ve karşı düşen VE-VEYA çizgesi (Russell & Norvig, 2002).

### 2.2.3 Geri yönde çıkarsamalar zinciri

Veri güdümlü bir yöntem olan ileri yönde çıkarsamalar zincirine alternatif olarak hedef güdümlü olan geri yönde çıkarsamalar zinciri ile de çıkarım yapılabilmektedir. Bu yöntemde doğruluğu test edilen ifadenin rekürsif bir şekilde öncüllerinin sağlanırlığı incelenerek, ilgili ifadenin bilgi tabanından elde edilip edilemeyeceği belirlenmeye çalışılır. Örnek olarak, Şekil 2.4'teki bilgi tabanında Q ifadesinin doğruluğu sorgulandığında, enine aramaya dayalı geri yönde çıkarsama zinciri ile 5 adımda sonuca ulaşılabilmektedir. Bu adımlar: $P \Longrightarrow Q$, $L \wedge M \Longrightarrow P$, $A \wedge P \Longrightarrow L$, $A \wedge B \Longrightarrow L$ ve $B \wedge L \Longrightarrow M$.

Geri yönde çıkarsamalar zinciri genelde hedef odaklı problemlerin çözümünde tercih edilir. Çoğu problemde bu yaklaşım, ileri yönde çıkarsamaya göre daha hızlı sonuca ulaşmaktadır ve yöntemin işlemsel karmaşıklığı bilgi tabanı boyutu ile doğru orantılı veya daha az olmaktadır.

# 3. UZAMSAL VE ZAMANSAL BİLGİNİN TEMSİL EDİLMESİ

Verilen bir ortam hakkında çıkarsama yapılabilmesi için uzamsal ve zamansal bilginin kullanımına ilişkin literatürde çeşitli niteliksel modeller mevcuttur. Uzamsal modeller, ortamdaki nesnelerin konumları üzerinden çıkarsama yapılırken kullanılan modellerdir. Zamansal modellerde ise ortamda gerçekleşen eylem ve olayların zamanları arasındaki ilişkiler ifade edilmektedir.

## 3.1 Uzamsal Modeller

Uzamsal çıkarsama için kullanılan modeller 2 grup altında toplanabilir: topolojik modeller ve yönsel modeller.

### 3.1.1 Topolojik modeller

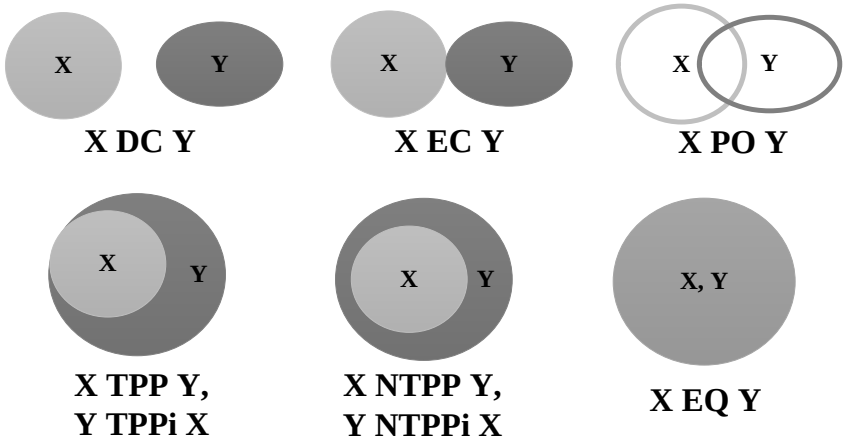
İki boyutlu ortamlarda, nesneler arasındaki topolojik ilişkiyi modelleyen iki temel yaklaşım bulunmaktadır: RCC8 (Randell, Cui, & Cohn, 1992) ve 9-kesişim analizi (Egenhofer & Herring, 1991).

#### 3.1.1.1 RCC8 bölge bağlantı analizi

RCC8 (*region connection calculus*) modelleme yaklaşımında, nesneler arasındaki ikili topolojik ilişkileri tanımlamak üzere Şekil 3.1'de görülen sekiz farklı durum söz konusu olabilir: ayrık (*DC – disconnected*), dıştan teğet (*EC – externally connected*), kesişen (*PO – partially overlapping*), teğet ve kapsanan (*TPP – tangential proper part*), teğet ve kapsayan (*TPPi – tangential proper part inverse*), teğet olmayan kapsanan (*NTPP – non-tangential proper part*), teğet olmayan kapsayan (*NTPPi – non-tangential proper part inverse*) ve eşit (*EQ – equal*).

#### 3.1.1.2 9-kesişim analizi

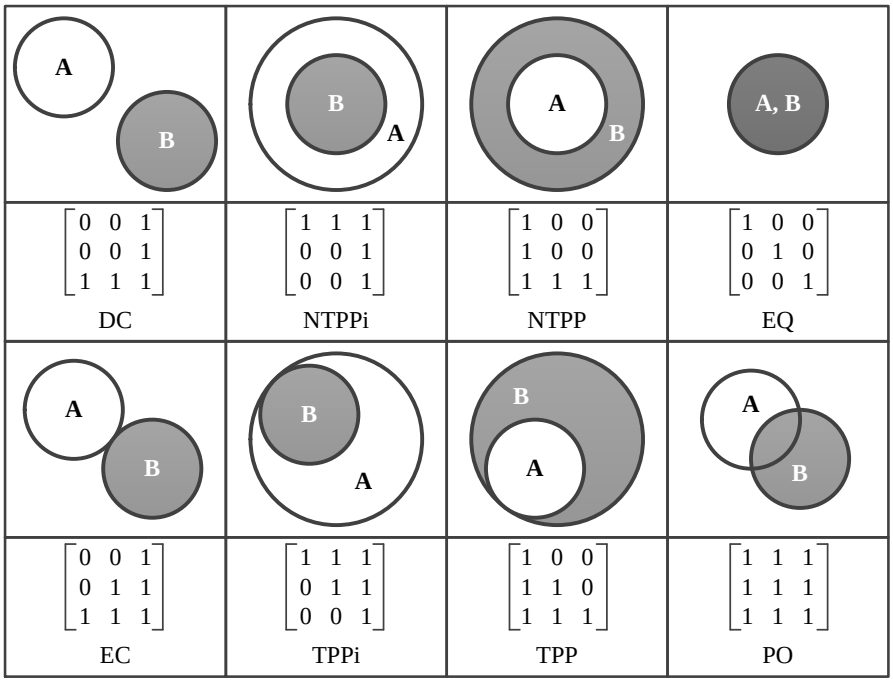
9-kesişim analizinde (*9-intersection calculus*) ikili topolojik ilişkileri modellemek üzere 3 özelliğin kesişiminden faydalanılmaktadır: iç ($X^o$ – *interior*), sınır ($\partial X$ – *boundary*) ve dış ($X^-$ – *exterior*). Bu temel özellikler arası kesişimlere göre ikili




**Şekil 3.1:** RCC8 modelindeki ilişkilerin geometrik gösterimi.

topolojik ilişkiler matrisi $M(A,B)$ oluşturulur (3.1). Bu matris, RCC8 modelindeki ikili ilişkileri biçimsel bir şekilde ifade etmektedir. Şekil 3.2'de farklı ikili topolojik ilişkilerin geometrik gösterimi için ilişkileri ifade eden (varsa 1, yoksa 0) matris görülmektedir. Örneğin, *DC* ilişkisinde nesneler ayrık olduğundan kesişimi ifade eden elemanlar $(M_{1,1}, M_{1,2}, M_{2,1}, M_{2,2})$ 0 olarak belirlenmiştir. Diğer tüm RCC8 ilişkileri için bu matris benzer şekilde temel kesişimlere göre oluşturulmaktadır.

$$M(A,B) = \begin{bmatrix} A^o \cap B^o & A^o \cap \partial B & A^o \cap B^- \\ \partial A \cap B^o & \partial A \cap \partial B & \partial A \cap B^- \\ A^- \cap B^o & A^- \cap \partial B & A^- \cap B^- \end{bmatrix} \quad \textbf{(3.1)}$$



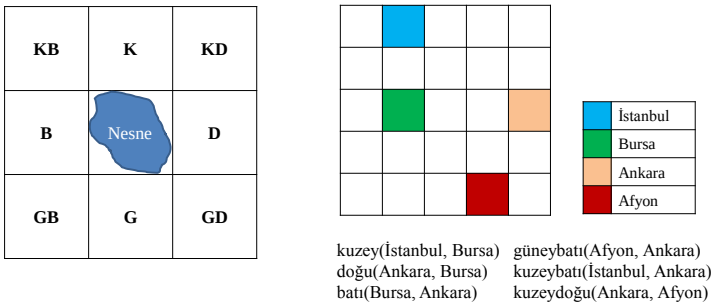

**Şekil 3.2:** 9-kesişim modelindeki ilişkilerin geometrik gösterimi.

### 3.1.2 Yönsel modeller

Temel olarak iki nesne arasındaki yönsel ilişkileri modellemede iki yöntem ön plana çıkmaktadır: ana yönler analizi (Frank, 1991) ve 2n-yıldız analizi (Mitra, 2002).

#### 3.1.2.1 Ana yönler analizi

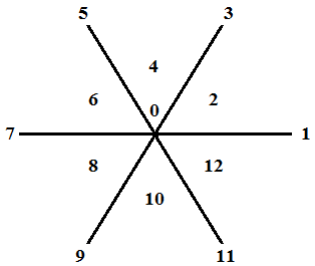
Ana yönler analizi (*cardinal direction calculus*) modeline göre, 2 boyutlu ortamdaki nesnelerin konumları arasındaki bağıl yönsel ilişkiler ana yönler cinsinden (kuzey, doğu, güney, batı, kuzeydoğu, güneydoğu, güneybatı ve kuzeybatı) ifade edilmektedir. Şekil 3.3 (a)'da bu modelde bir nesnenin çevresindeki bölgelerin yönlere göre ayrımı verilmiştir. Şekil 3.3 (b)'de ise bu modeldeki ikili ilişkiler bir örnek üzerinde gösterilmektedir. Bu örnekte, "İstanbul Bursa'nın kuzeyindedir.", "Ankara Bursa'nın doğusundadır.", "Afyon Ankara'nın güneybatısındadır." gibi iki şehre karşı düşen bölgeler arasındaki yönsel ilişkiler ifade edilmektedir.




**Şekil 3.3:** Ana yönler analizindeki (a) ikili ilişkiler ve (b) bu ilişkilere örnekler.

#### 3.1.2.2 2n-yıldız analizi

2-n yıldız analizinde (*2n-star calculus*) ana yönler herhangi bir $n > 0$ değeri için bölge açısı $\frac{360}{n}$ olacak şekilde genellenerek $2n + 1$ adet ikili ilişki tanımlanmaktadır. Örnek olarak 12-yıldız modelindeki 13 farklı ilişki Şekil 3.4'te verilmiştir.



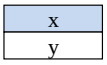
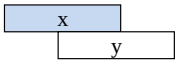
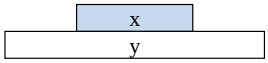
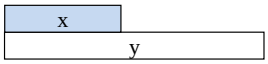
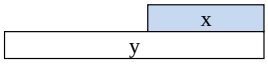

**Şekil 3.4:** 12-yıldız modelindeki 13 yönsel ilişkinin geometrik gösterimi.

### 3.2 Zamansal Modeller

Zaman bilgisi üzerinden çıkarım yapılırken kullanılan belli başlı modeller: zaman aralığı cebri (Allen, 1983), nokta cebri (Vilain, Kautz, & van Beek, 1989), nokta-zaman aralığı cebri (Vilain M. B., 1982) ve INDU analizi (Pujari, Kumari, & Sattar, 1999) şeklinde sıralanabilir.

#### 3.2.1 Zaman aralığı cebri

Allen'in zaman aralığı cebrinde (*Allen's interval algebra*) olaylara ilişkin zaman aralıkları arasında 13 adet ilişki tanımlanmaktadır. Şekil 3.5'te gösterilen bu ilişkiler yardımıyla olaylara karşı düşen zaman aralıkları arasında, bir olayın diğerinden önce ($b$) ya da sonra ($bi$) olması, bir olayın diğerini başlatması ($s$) ya da sonlandırması ($f$), eş zamanlı olaylar ($eq$), olay zamanlarında kesişim ($o$ ve $oi$) ve kapsama ($d$ ve $di$) gibi olası tüm farklı durumlar modellenebilmektedir. Bu ilişkiler üzerinde çıkarsama yapılırken bütün olaylar arasındaki ikili ilişkileri barındıran en küçük çizge oluşturulur. Bu çizgeyi oluşturmanın karmaşıklığı NP-zordur (*NP-hard*).

| İlişki | Sembol | Tersi | Gösterimi |
|---|---|---|---|
| X before Y<br>Y after X | b | bi |  |
| X equal Y | eq / = | |  |
| X meets Y<br>Y met-by X | m | mi |  |
| X overlaps Y<br>Y overlapped-by X | o | oi |  |
| X during Y<br>Y contains X | d | di |  |
| X starts Y<br>Y started-by X | s | si |  |
| X finishes Y<br>Y finished-by X | f | fi |  |

**Şekil 3.5:** Allen'in zaman aralığı cebrindeki ilişkiler.

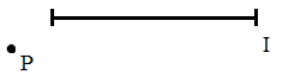
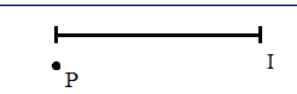
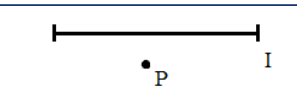
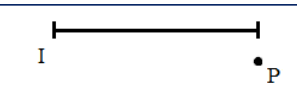
#### 3.2.2 Nokta cebri

Nokta cebrinde (*point algebra*) olaylar zamanda noktalarla temsil edilmekte ve bu noktalar arasında 3 temel ilişki kurulmaktadır: $<$, $=$ ve $>$. Bu zamansal modelleme

yaklaşımında, Allen'in zaman aralığı cebrine göre daha az sayıda ilişki tipi kullanıldığından, tüm olaylar arasındaki ikili ilişkiler üzerinde çıkarsama yapmanın karmaşıklığı daha azdır ($O(n^3)$). Fakat bu yöntem zamansal ilişkileri belirtmede zaman aralığı cebri kadar betimleyici değildir. Örneğin, bir olayın diğerini başlatması ya da durdurması gibi bazı ilişkiler bu yaklaşımla modellenemez.

### 3.2.3 Nokta-zaman aralığı cebri

Nokta-zaman aralığı cebrinde (*point-interval algebra*) olaylar anlık ve süreli olaylar şeklinde ayrılmakta ve anlık olaylar noktalarla temsil edilirken, süreli olayların temsilinde zaman aralıkları kullanılmaktadır. Allen'in zaman aralığı cebrine benzer şekilde olaylar arasında 13 ilişki tanımlanmıştır. Bu ilişkiler Şekil 3.6'da verilmiştir.

| İlişki | Gösterimi |
|---|---|
| *P1 before P2* |   |
| *P2 after P1* | |
| *P1 equals P2* |   |
| *P before I* |   |
| *I after P* | |
| *P begins I* |   |
| *I begun-by P* | |
| *P during I* |   |
| *I contains P* | |
| *P ends I* |   |
| *I ended-by P* | |
| *P after I* |   |
| *I before P* | |

**Şekil 3.6:** Nokta-zaman aralığı cebrindeki ilişkiler.

### 3.2.4 INDU analizi

INDU analizinde (*INDU calculus*), Allen'in zaman aralığı cebrindeki ilişkilerde karşılaştırılan olayların süreleri arasındaki bağıl ilişkiler de eklenerek 25 temel ilişki tanımlanmıştır. Şekil 3.5'te belirtilen $eq$, $s$, $si$, $d$, $di$, $f$ ve $fi$ sembolleri ile ifade edilen 7 ilişkide zaman aralıklarının süreleri arasındaki ilişkiler belli olduğundan bunlar aynen alınmış, diğer 6 ilişki ($b$, $bi$, $m$, $mi$, $o$ ve $oi$) içinse süreler arasındaki olası 3 durum ($<$, $=$ veya $>$) göz önüne alınarak 18 ilişki tanımlanmıştır.

# 4. SEMBOLİK PLANLAMA

## 4.1 Özerk Eylem Planlama

Otonom etmenler belirli bir ortamda verilen çeşitli görevleri yerine getirmek üzere tasarlanırlar. Verilen bir görevin yerine getirilebilmesi için, etmenin ortamın mevcut durumu ile hedeflenen durum arasında hangi eylemleri hangi sırada gerçekleştirmesi gerektiğini önceden belirlemesi gerekir. Bu sürece özerk eylem planlama adı verilir (Russell & Norvig, 2002). Otonom olarak planlama yapan bir etmen sahip olduğu biçimsel ortam modeline göre ortamda gerçekleştirebildiği eylemleri önkoşul ve beklenen etkileriyle birlikte göz önüne alarak ortamın durumunu hedeflenen duruma getirecek eylemlerin sıralı dizisini bulur.

## 4.2 Planlama Ortamının Modellenmesi

Otonom etmenlerin planlama yapabilmeleri için ortama ve probleme ilişkin bilgilerin biçimsel ifadelerine sahip olmaları gereklidir. Planlama yöntemlerinin etkin bir şekilde hedefe yönelik bir çözüm üretebilmesi planlama ortam modelinin eksiksiz ve doğru olarak hazırlanmasına bağlıdır. Planlama ortam modeli, problem tanımı ve ortamda etmen tarafından yürütülebilen eylemlere ilişkin önkoşul ve etkilere ait bilgilerden oluşur. Planlama ortamı temsil edilirken genel olarak plan operatörleri, ortamdaki nesneler ve bu nesnelerin mevcut durumunu belirten ifadeler kullanılır.

### 4.2.1 Planlama dilleri

Klasik planlamada, ortamın modellenmesi ve problemin temsilinde faydalanılan 3 temel dilden bahsedilebilir. Bunlar,

- STRIPS (Stanford Research Institute Problem Solver)
- ADL (Action Description Language)
- PDDL (Planning Domain Definition Language) dilleridir.

Bu dillerden en eski ve temel olanı STRIPS planlama dilidir (Fikes & Nilsson, 1971). STRIPS, kendi eylemleri üzerinde çıkarsama yapabilen ilk gezgin robot olarak bilinen Shakey (Şekil 4.1) için geliştirilmiştir (SRI International, 2012). Bu dilde ortamın mevcut durumu pozitif önermeler (*proposition*) kullanılarak temsil edilir ve problemin hedef koşulları da durum temsilinde kullanılan önermelerle gösterilir. Hedef tanımında yalnızca hedef koşulunu sağlamada gerekli olan önermeler verilir. Plan operatörlerinde önkoşullar pozitif önermelerle ifade edilirken, etkiler ise ortamın mevcut durumuna eklenecek pozitif önermeler ve ortamdan silinecek negatif (tümlenmiş) önermelerle temsil edilir.

**Şekil 4.1:** Shakey robotu (SRI International, 2012).

STRIPS dilinin genişletilmiş bir versiyonu olan ADL planlama diliyle daha karmaşık ortamlar ve problemler modellenebilmektedir (Pednault, 1989). Bu planlama dilinde planlama domeni ve problem temsil edilirken, STRIPS diline ek olarak, negatif önkoşullar, denklik ve eşitsizlik koşulları, mantıksal VEYA operatörü (∨) ve niceleyici operatörler (∀ ve ∃) kullanılabilmektedir.

PDDL dili plan domeni ve probleminin temsili konusunda standart bir dil ortaya koyma amacıyla oluşturulmuş bir planlama dilidir (McDermott, et al., 1998). Bu dil, özellikle, zamansal planlama problemlerinin ifade edilebilirliğini artırmaktadır. STRIPS, ADL ve PDDL dillerinin özellik ve farklılıklarına ilişkin bir karşılaştırma Çizelge 4.1'de gösterilmektedir.

**Çizelge 4.1:** STRIPS, ADL ve PDDL dillerinin karşılaştırılması (Russell & Norvig, 2002; Kalisch & König, 2005).

| **STRIPS** | **ADL** | **PDDL** |
|---|---|---|
| Kapalı ortam varsayımı: Bahsedilmeyen önermeler yanlış kabul edilir. | Açık ortam varsayımı: Bahsedilmeyen önermeler bilinemez kabul edilir. | Kapalı ortam varsayımı: Bahsedilmeyen önermeler yanlış kabul edilir. |
| Durumlar sadece olumlu önermelerle gösterilir. | Durumlar olumlu ve olumsuz önermelerle gösterilebilir. | Durumlar olumlu ve olumsuz önermelerle gösterilebilir. |
| $A \wedge \neg B$: $A$'yı ekle, $B$'yi çıkar. | $A \wedge \neg B$: $A$ ve $\neg B$'yi ekle, $\neg A$ ve $B$'yi çıkar. | $and(A\ not\ (B))$: $A$ ve $\neg B$'yi ekle, $\neg A$ ve $B$'yi çıkar. |
| Hedef ifadesi $\wedge$'lerden oluşur, $\vee$ içermez. | Hedef ifadesi $\wedge$ ve $\vee$'lardan oluşabilir. | Hedef ifadesi $\wedge$ ve $\vee$'lardan oluşabilir. |
| Hedef temsil edilirken her ($\forall$) ve bazı ($\exists$) operatörleri kullanılmaz. | Hedef temsil edilirken her ($\forall$) ve bazı ($\exists$) operatörleri kullanılabilir. | Hedef temsil edilirken her ($\forall$) ve bazı ($\exists$) operatörleri kullanılabilir. |
| Koşullu etkiler yoktur. | Koşullu etkiler tanımlanabilir. | Koşullu etkiler tanımlanabilir. |
| Eşitlik ($=$) ve eşitsizlik ($\neq$) ifadelerini desteklemez. | Eşitlik ($=$) ve eşitsizlik ($\neq$) ifadelerini destekler. | Eşitlik ($=$) ve eşitsizlik ($\neq$) ifadelerini destekler. |
| Değişkenlerin tipleri yoktur. | Değişkenlerin tipleri vardır. | Değişkenlerin tipleri vardır. |

### 4.2.2 Planlama probleminin temsili

Örnek bir planlama problemi olarak maymun-muz problemi verilebilir. Bu problemde bir odada aç bir maymun ve tavana asılı bir demet muz bulunmaktadır. Maymunun amacı bu muzları elde etmektir. Ortamdaki nesneler, maymun, muz, kutu ve zemin olarak düşünüldüğünde bu problemin başlangıç ve hedef durumları Şekil 4.2'deki gibi yüklem mantığıyla (*first-order logic*) modellenebilir.

| | |
|---|---|
| *üzerinde(maymun, zemin),*<br>*üzerinde(kutu, zemin),*<br>*konum(maymun, a),*<br>*konum(kutu, b),*<br>*konum(muz, c),*<br>*durum(muz, tavana_asılı).* | *durum(muz, alındı).* |
| **(a)** | **(b)** |

**Şekil 4.2:** Maymun-muz probleminin (a) başlangıç ve (b) hedef durumları.

### 4.2.3 Planlama domeninin temsili

Planlama domeninde otonom etmenin verilen görevleri yerine getireceği ortamın modeline ilişkin biçimsel ifadeler tanımlanır. Bu model oluşturulurken, etmenin ortamda gerçekleştirebileceği eylemlerin önkoşul ve etkileri gerçek dünya problemindeki olası değişik durumlar göz önüne alınarak ifade edilir. Eylemlerin ortamda neden olacağı durum geçişlerini koşul bilgisiyle beraber barındıran temsile plan operatörleri adı verilir.

Maymun-muz problemindeki plan operatörlerinin STRIPS planlama dilindeki temsilleri Çizelge 4.2'de verilmiştir. Maymunun ortamda gerçekleştirebildiği eylemlere karşı düşen bu operatörlerden *git(X,Y)* maymunun oda içinde konum değiştirmesini, *taşı(kutu,X,Y)* maymunun kutuyu oda içinde *X* konumundan *Y* konumuna taşımasını, *tırman(kutu)* maymunun kutu üzerine tırmanmasını ve *al(muz)* maymunun muzu almasını modeller. Her bir operatörün önkoşulları ilgili operatörün uygulanabilmesi için ortamda sağlanması gereken önermeleri ifade eder. Bir operatöre karşı düşen eylem yürütüldüğünde, o operatörün silme listesindeki ifadeler ortamın mevcut durumundan çıkarılır ve ekleme listesindeki ifadeler ortamın mevcut durumunu belirten önermeye eklenir. Örneğin, *taşı(kutu,$k_1$,$k_2$)* operatörünün yürütülebilmesi için maymun ve kutunun $k_1$ konumunda bulunması gereklidir. İlgili eylem başarıyla yürütüldüğü takdirde, maymun ve kutunun yürütme öncesi konumuna ilişkin ortam durumundaki ifadeler tümlenir ve işlem sonundaki konum bilgilerine ($k_2$) ilişkin ifadeler ortama eklenir.

**Çizelge 4.2:** Maymun-muz problemine ilişkin plan operatörleri.

| Operatör | Önkoşullar | Silme Listesi | Ekleme Listesi |
|---|---|---|---|
| *git(X,Y)* | *konum(maymun,X), üzerinde(maymun,zemin)* | *konum(maymun,X)* | *konum(maymun,Y)* |
| *taşı(kutu,X,Y)* | *konum(maymun,X), konum(kutu,X), üzerinde(maymun,zemin), üzerinde(kutu,zemin)* | *konum(maymun,X), konum(kutu,X)* | *konum(maymun,Y), konum(kutu,Y)* |
| *tırman(kutu)* | *konum(maymun,X), konum(kutu,X), üzerinde(maymun,zemin), üzerinde(kutu,zemin)* | *üzerinde(maymun,zemin)* | *üzerinde(maymun,kutu)* |
| *al(muz)* | *üzerinde(maymun,kutu), konum(kutu,X), konum(muz,X), durum(muz,tavana_asılı)* | *durum(muz,tavana_asılı)* | *durum(muz,alındı)* |

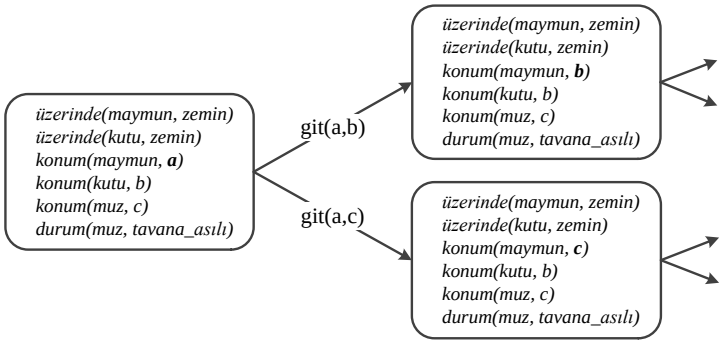
### 4.3 Klasik Planlama Yöntemleri

Klasik planlama probleminde, planlama domeni ve problem modeli oluşturulduktan sonra, çeşitli yöntemlerden faydalanılarak çözüme yönelik planın otonom bir şekilde üretilmesi sağlanabilir. İlk akla gelen temel planlama yöntemleri: ileri durum uzayı arama (*progression planning*) ve geri durum uzayı arama (*regression planning*) şeklinde verilebilir. Bunun yanı sıra, kısmi-sıralı planlama (*partial-order planning*), planlama çizgesi üzerinden planlama (*Graphplan*) ve hiyerarşik görev ağı planlama (*hierarchical task network planning*) yöntemleri de sık kullanılan temel klasik planlama yöntemlerindendir.

#### 4.3.1 İleri durum uzayı arama

Bu yöntemde planlamaya ortamın ilk durumundan başlanarak hedeflenen duruma ulaşmaya yönelik bir eylem dizisi bulunmaya çalışılır. Bu amaçla çeşitli arama yaklaşımlarından faydalanılabilir (Genişlik öncelikli arama, Derinlik öncelikli arama, Düşük maliyetli arama, $A^*$ arama). Her iterasyonda, etmenin ortamda uygulayabileceği (önkoşulları mevcut durumda sağlanan) bir eylem seçilerek o eylemin etkileri ile durum güncellemesi yapılır ve uygulanabilir diğer eylemler de arama ağacında tutulur. Arama, hedef durumun koşullarını sağlamak için gereken bütün durum ifadelerinin ortamda mevcut olduğu bir hedef testi fonksiyonu ile teyit edilene kadar devam eder.

Şekil 4.3'te ileri durum uzayı arama yöntemi kullanılarak, maymun-muz probleminin ilk durumunda (Şekil 4.2) uygulanabilen plan operatörleri (Çizelge 4.2) verilmiştir. Bu durumda, *git(a,b)* eylemi yürütüldüğünde maymunun konumu *b* olarak güncellenmekte, *git(a,c)* eylemi seçildiğinde ise maymunun konumu *c* olmaktadır. İleri yönde arama, hedeflenen duruma ulaşılana kadar benzer şekilde devam eder.



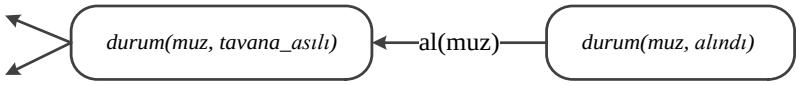

**Şekil 4.3:** İleri durum uzayı arama örneği.

Bu yöntemdeki problem, ilk durumdan itibaren uygulanabilen tüm eylemler göz önüne alınarak arama yapılmasıdır. Genelde, herhangi bir durumda önkoşulları sağlanan pek çok çözümle ilgisi olmayan eylem bulunabileceği için arama uzayının boyutu hedef duruma ulaşmayı güçleştirecek seviyede büyüyebilir. Bu sorunu aşmak için çoğu zaman çeşitli buluşsallardan (*heuristic*) yararlanılmaktadır.

### 4.3.2 Geri durum uzayı arama

Bu yöntemde ileri durum uzayı arama yönteminin aksine hedef durumdan ilk duruma ulaşacak şekilde arama yapılır. Her iterasyonda, ortamın mevcut durumuna ulaşırken yürütülmesi mümkün olan (etkileri mevcut durum ifadesinde kapsanan) bir eylem seçilerek o eylemin önkoşullarını sağlayacak şekilde geri yönde durum güncellenir. Geri durum uzayı aramada, eylem seçiminde tutarlılık (*consistency*) önemlidir. Seçilen bir eylemin tutarlı olabilmesi için, etkilerinin hedefe yönelik olmasının yanı sıra mevcut olumlu etkileri geri almaması da istenir.

Şekil 4.4'te geri durum uzayı arama yöntemi kullanılarak, maymun-muz probleminin hedef durumuna (Şekil 4.2) ulaşırken uygulanabilen plan operatörleri (Çizelge 4.2) verilmiştir. Hedef durum olan *durum(muz, alındı)* etkisine sahip tek operatör *al(muz)* olduğundan, bu operatörün önkoşulları göz önüne alınarak bir önceki durum belirlenir. Geri yönde arama benzer şekilde, ortamın ilk durumuna ulaşılıncaya kadar devam eder.




**Şekil 4.4:** Geri durum uzayı arama örneği.

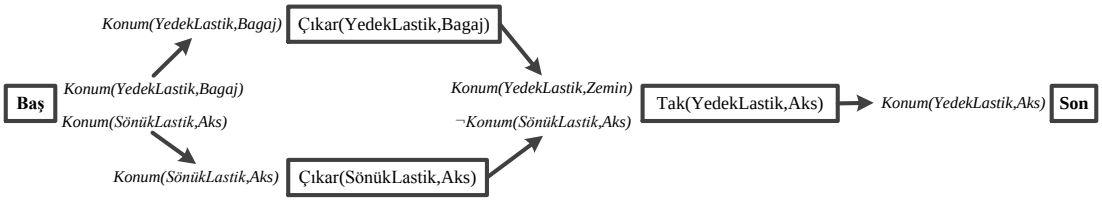
Hedef odaklı bu yöntemde, arama ağacına alınan tüm eylemler hedefe ulaşmada kullanılabilen eylemler olduğundan dallanma faktörü ileri durum uzayı arama yöntemine göre daha azdır ve çözüme ulaşmak çoğu zaman daha kolay ve hızlıdır.

### 4.3.3 Kısmi-sıralı planlama

Durum uzayı planlama yöntemleri ile elde edilen eylemlerin tam sıralı olduğu planlardan farklı olarak, bu yöntemde çözüme yönelik olarak yürütülmesi gereken eylemlerin mümkün olduğunca sıralama kısıtı olmayacak bir şekilde belirlenmesine çalışılır. Aramaya hedef durumdan başlanır ve geri yönde arama yapılarak mevcut

durumla uygun etki üreten eylemlerin önkoşulları sağlanmaya çalışılır. Eylemlerin plana eklenmesi sırasında en az bağımlılık (*least commitment*) stratejisi izlenerek, eklenen eylemin mümkün olduğunca diğer eylemlerle sıra ilişkisi olmamasına özen gösterilir. Daha önce eklenmiş operatörlerin önkoşullarını oluşturan etkiye sahip eylemler eklendiğinde ya da aykırı durum oluştuğunda sıralama kısıtları eklenir. Arama işlemi planda kullanılan her bir eylemin önkoşulları sağlanana kadar veya eylemler arasında çatışmaya sebep olan aykırı durumlar oluştuğu sürece devam eder.

Şekil 4.5'te havası yetersiz araba lastiğinin yedek lastikle değiştirilmesi işlemine ait kısmi sıralı plan görülmektedir. Bu işlem sırasında uygulanabilen 3 eylem bulunmaktadır: değiştirilecek lastiğin akstan sökülmesi (*Çıkar(SönükLastik, Aks)*), yedek lastiğin bagajdan çıkarılması (*Çıkar(YedekLastik, Bagaj)*) ve yedek lastiğin ilgili aksa takılması (*Tak(YedekLastik, Aks)*). Havası yetersiz olan lastiğin sökülmesi ile bagajdan yedek lastiğin çıkarılmasına ilişkin eylemlerinin şekilde görülen önkoşul ve etkileri bu eylemler arasına sıralama kısıtı eklenmesini gerektirmediğinden, bu iki eylem aynı anda gerçekleştirilebilmektedir.




**Şekil 4.5:** Kısmi sıralı plan örneği (Russell & Norvig, 2002).

### 4.3.4 Planlama çizgesi üzerinden planlama

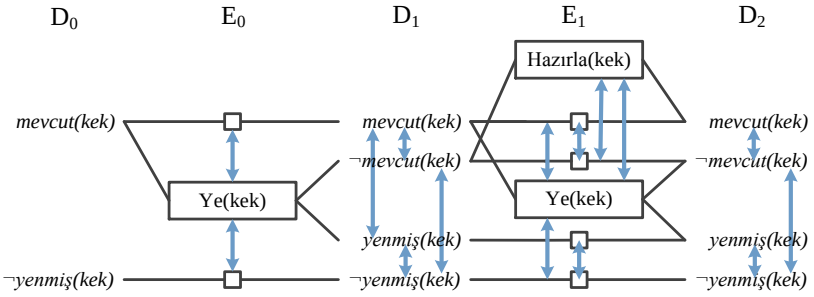
Planlama çizgesi veri yapısının oluşturulması ve analiz edilmesi mantığına dayanan Graphplan, STRIPS diliyle kodlanmış domenler üzerinde sembolik planlama işleminde ve çözüme yönelik buluşsalların (*heuristic*) üretilmesinde kullanılmaktadır (Blum & Furst, 1995). Bu yöntemde zaman ekseninde seviyeler oluşturulur ve her bir seviyede hangi eylemlerin gerçekleştirilebileceğine karar verilir. Birbirleri ile etki ya da önkoşulları nedeniyle çelişen eylemler elenerek arama uzayı daraltılmaya ve böylece problemin esnetilmiş (hedef duruma ulaşmada gereken bir takım kısıtların ihmal edildiği) şekline bir çözüm bulunmaya çalışılır. Bulunan çözüm asıl problemde hedefe yönelik sembolik planlama sürecinde buluşsal olarak kullanılabileceği gibi, bu çözüme ulaşırken kullanılan eylemler göz önüne alınarak daraltılmış arama uzayında geri yönde aramayla hedef durumu sağlayan bir plan da üretilebilir.

Planlama çizgesi kullanımına örnek olarak, Şekil 4.6'da kek problemine ilişkin problem ve domen tanımları verilmiştir. İlk durumda, ortamda bir adet kek bulunmaktadır. Hedeflenen durumda ortamda hem kekin mevcut olması, hem de etmenin kek yemiş olması amaçlanmaktadır. Bu amaca yönelik olarak, etmenin ortamda gerçekleştirebileceği iki eyleme ilişkin plan operatörleri verilmiştir. Bunlardan *Ye(kek)* eylemi ile ortamda mevcut olan kek etmen tarafından yenerek tüketilmektedir. *Hazırla(kek)* eylemi ise ortamda kek bulunmadığında, kekin hazırlanarak ortamda mevcut duruma getirilmesi için kullanılabilmektedir.

| **İlk durum:** | **Plan operatörleri:** |
|---|---|
| *mevcut(kek)* | *Ye(kek):*<br>önkoşul: *mevcut(kek)*<br>etki: ¬*mevcut(kek)* ∧ *yenmiş(kek)* |
| **Hedeflenen durum:**<br>*mevcut(kek)* ∧ *yenmiş(kek)* | *Hazırla(kek):*<br>önkoşul: ¬*mevcut(kek)*<br>etki: *mevcut(kek)* |
| **(a)** | **(b)** |

**Şekil 4.6:** Kek probleminin (a) sembolik tanımı ve (b) plan operatörleri (Russell & Norvig, 2002).

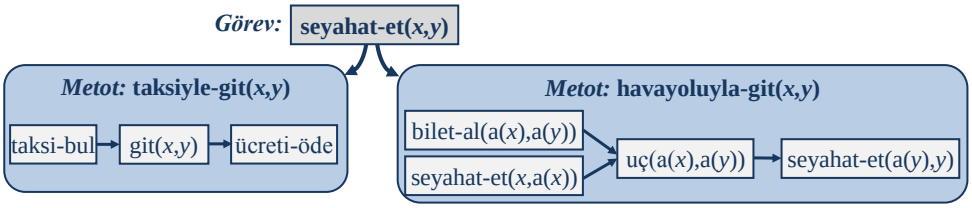
Şekil 4.6'da sembolik tanımı verilen probleme ilişkin planlama çizgesi Şekil 4.7'de verilmiştir. Planlama çizgesi, problemin ilk durumundan itibaren seviyeler halinde, her bir seviyede uygulanabilen eylemlerin etkileri göz önüne alınarak oluşturulur. Ardışık iki durum seviyesindeki ifadeler birbiri ile aynı olduğunda çizge oluşturma işlemi tamamlanır. Şekildeki, çift yönlü oklar ifade ve eylemler arasındaki karşılıklı dışlamaları (*mutex*) göstermektedir. Aynı seviyedeki eylemler ve durum ifadeleri arasında tanımlanan bu karşılıklı dışlamalar, etkileri veya önkoşulları çelişen eylemler (örneğin, *Hazırla(kek)* ve *Ye(kek)*), birbiriyle çelişen ifadeler (örneğin, *mevcut(kek)* ve ¬*mevcut(kek)*) ve etkileri veya önkoşulları çelişen eylem-ifade çiftleri (örneğin, *Ye(kek)* ve *mevcut(kek)*) aralarında tanımlanmaktadır.




**Şekil 4.7:** Kek problemine ilişkin planlama çizgesi (Russell & Norvig, 2002).

### 4.3.5 Hiyerarşik görev ağı planlama

Bu planlama yaklaşımında verilen yüksek seviyeli görevler, hiyerarşik olarak daha basit alt görevlerle ifade edilmeye çalışılır. Verilen tüm görevler, etmen tarafından gerçekleştirilebilen ve birbirleri ile uyuşmazlık problemi olmayan ilkel eylemlere indirgendiğinde planlama işlemi tamamlanmış olur. Buna örnek olarak Şekil 4.8'de seyahat etme görevinin alt metotlara ayrılması verilmiştir. Bu işlemde, iki şehir arasında gerçekleştirilecek seyahat görevi, taksi kullanmak veya hava yoluyla gitmek şeklinde iki alternatif metot ile ifade edilmiştir. Daha sonra bu metotlar da kendi içinde sembolik plan operatörleri ile temsil edilmiştir. Bu metotlara mesafeye göre maliyet eklenerek kısa mesafelerde taksinin, uzun mesafelerde ise uçağın tercih edilmesi sağlanabilir.




**Şekil 4.8:** Hiyerarşik görev ağı ile planlama örneği (Nau, 2009).

## 4.4 Planlama Operatörlerinin Öğrenilmesi

Planlama operatörleri tümüyle tanımlı olmadığında, etmenlerin ortamda gerçekleştirebildikleri eylemlerin önkoşul ve etkilerini ortamdan gözlemlerle öğrenmeleri gerekmektedir. Bu amaçla çeşitli yaklaşımlar geliştirilmiştir. Bunlardan EXPO robotların planlama operatörlerindeki hata ve eksiklikleri gerçek dünyadan elde edilen gözlemlerle öğrenmesine ve düzeltmesine dayalı bir yöntemdir (Gil, 1994). Wang'ın OBSERVER adlı sisteminde ise, planlama operatörleri, başka robotların eylemlerinin oluşturduğu durum değişimlerinin gözlemlenmesi ve gözlemlerden elde edilen bilginin ortamda sınanması ile öğrenilmektedir (Wang, 1995). Walsh ve Litman'ın çalışmasında web hizmetlerinde öğrenme problemi STRIPS eylem modellerinin öğrenilmesi şeklinde incelenmiştir (Walsh & Littman, 2008). Bu üç yöntemin ortak özelliği eylemler arasındaki durumların tam gözlemlenebilir olduğu varsayımına dayanmalarıdır.

Yakın zamanda geliştirilen yöntemler ise daha az bilgi kullanılarak eylem temsillerinin öğrenildiği kısmi gözlemlenebilir ortamlarda çalışabilen yöntemlerdir.

Bunlardan en dikkat çekici olanları SLAF (Shahaf & Amir, 2006; Shahaf, Chang, & Amir, 2006; Amir & Chang, 2008), ARMS (Yang, Wu, & Jiang, 2005; Wu, Yang, & Jiang, 2007), LAMP (Zhuo, Yang, Hu, & Li, 2010), OPMAKER (McCluskey, Richardson, & Simpson, 2002), OPMAKER2 (McCluskey, Cresswell, Richardson, & West, 2009), LOCM (Cresswell, McCluskey, & West, 2009; Cresswell, McCluskey, & West, in press) ve LOCM2'dir (Cresswell & Gregory, 2011).

SLAF, kısmen gözlemlenebilir ortamlarda eylemlerin önkoşul ve etkilerinin belirlenmesine yönelik mantıksal filtrelemeye dayalı bir yaklaşımdır (Shahaf & Amir, 2006; Shahaf, Chang, & Amir, 2006; Amir & Chang, 2008). Bu yöntemde öğrenilen bilgilerin doğrulanması amacıyla ortamda gözlemlenen ara durumlardan faydalanılmaktadır.

ARMS, SLAF'den farklı olarak eylemler arasındaki ortama ilişkin durum bilgisini kullanmadan, sadece gözlemlenen eylemler ve ortamın ilk durumuna ilişkin bilgiden yararlanarak STRIPS plan operatörlerini öğrenebilen etkin bir yaklaşımdır (Yang, Wu, & Jiang, 2005; Wu, Yang, & Jiang, 2007). Bu yöntem, ortak parametreler üzerinden birbiriyle ilişkili olan eylem ve ifade kümeleri arasında ağırlıklı maksimum sağlanabilirlik gözeterek eylemlerin önkoşul ve etkilerini belirlemeye dayanmaktadır. Bu yaklaşımın bir dezavantajı koşullu etkilerin öğrenilememesidir.

LAMP, niceleyiciler ve mantıksal gerektirmeler içeren daha karmaşık eylem modellerini öğrenmeye olanak veren bir yöntemdir (Zhuo, Yang, Hu, & Li, 2010). Markov mantıksal ağlarını (Richardson & Domingos, 2006) kullanan bu yöntemde, plan örneklerinden PDDL dilindeki eylem temsiline uygun örüntüler (önkoşul, etki ve koşullu etkileriyle) çıkarılmaktadır. Bu yöntemde de ara durumların kullanımı hata oranını kabul edilebilir seviyelerde tutmak için gereklidir.

Plan operatörlerinin elle hazırlanması sürecinin zorluğunu gidermek için ortaya atılan OPMAKER ise, ortamın ilk durumuna ilişkin mantıksal ifadeler ve örnek eylem dizileri kullanılarak plan operatörlerinin parametreleri, önkoşulları ve etkilerinin belirlenmesine yönelik bir yaklaşımdır (McCluskey, Richardson, & Simpson, 2002). Bu yöntemle plan operatörlerinin öğrenilmesi için ortamdaki nesneler, nesnelerin sınıf bilgileri ve ortamı ifade etmede kullanılan mantıksal ifadeleri kapsayan kısmi domen modeli sisteme girdi olarak verilmelidir. Ayrıca, eylemler arasındaki durum bilgisine ilişkin kısmi gözlemler de öğrenme aşamasında kullanılmaktadır.

OPMAKER2 sisteminde ise eylemler arasındaki ortam durumuna ilişkin gözlemlere ihtiyaç duyulmamaktadır (McCluskey, Cresswell, Richardson, & West, 2009). Bununla birlikte, nesneler ve nesne gruplarına ilişkin kısmi domen modeli, ortamın ilk durumuna ilişkin bilgi ve plan örnekleriyle beraber sisteme girdi olarak verilmelidir.

Diğer tüm yöntemlerden farklı olarak LOCM, sadece ortamda gözlemlenen eylemlerin sıralı dizisini kullanarak plan operatörlerini öğrenebilmektedir (Cresswell, McCluskey, & West, 2009; Cresswell, McCluskey, & West, in press). Ayrıca, nesne-tabanlı olan bu yöntemle ortamdaki nesnelerin durum değişimlerini de sonlu durum makineleriyle modellemek mümkündür. Bu nedenle LOCM bu çalışmanın ilk aşamasında kullanılabilecek uygun bir yöntem olarak belirlenmiştir.

LOCM2, LOCM'nin farklı davranışlar gösteren nesnelerin modellenmesindeki durum genelleme sorununu çözmek için ortaya atılmış bir yaklaşımdır (Cresswell & Gregory, 2011). Bu yöntemde, örnek planlar durum geçişi tabanlı bir temsille analiz edilmekte ve LOCM'nin varsayımlarındaki kısıtlamalar değişken davranışlar gösteren nesne grupları için esnetilerek bu tip nesnelerin birden fazla sonlu durum makinesi ile modellenmesine imkan verilmektedir. Durum geçişlerinde ardışıklık analizine dayalı olan bu yöntemin LOCM'ye göre dezavantajı, bu yöntemle sonlu durum makinelerinin doğrudan elde edilememesidir.

# 5. PROBLEM TANIMI VE ÖNERİLEN SİSTEM

## 5.1 Giriş

Bu bölümde, tez çalışması kapsamında incelenen problemin tanımı verilmekte ve problemin çözümüne yönelik olarak geliştirilen iki aşamalı sistem genel olarak tanıtılmaktadır.

## 5.2 Problem Tanımı

Çalışma ortamı hakkında kısıtlı bilgiye sahip olan bir etmenin, verilen görevleri başarıyla yerine getirebilmesi için ortamı gözlemlerle tanıması gerekmektedir. Benzer şekilde, tez çalışması kapsamında üzerinde durulan problemde, TIM ortamındaki olaylar ve nesneler arasındaki ilişkilerin otonom bir şekilde elde edilmesi amaçlanmaktadır. Bu amaçla, ortamdaki nesneler üzerinde gerçekleşen olayların bir listesini barındıran metin tabanlı eğitim senaryoları kullanılmaktadır. Sisteme girdi olarak verilen bu eğitim senaryoları, nesneler üzerinde gerçekleşen olayların sıralı dizisi ($\mathcal{O}$), ortamda farklı yönelimlerde bulunabilen nesnelere ilişkin yönelim bilgisi ($\mathcal{Y}$) ve nesneler arası ilişkileri barındıran bir bilgi tabanından ($\mathcal{I}$) oluşmaktadır. Eğitim senaryoları dışında, ortamdaki olayların önkoşul ve etkileri, nesneler ve aralarındaki etkileşime ilişkin bilgi bulunmamaktadır.

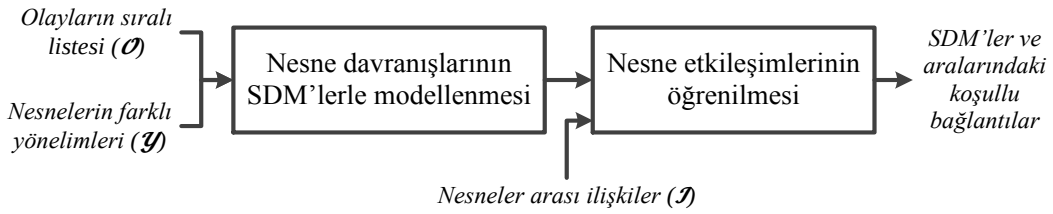
Olaylar sisteme gerçekleşme sırasına göre sıralanmış bir liste halinde verilmektedir ve bu listede eş zamanlı olaylar (eğer varsa) birbirini takip eden sıralarda yer almaktadır $\left(\forall i,j\ i<j\ \Rightarrow O_i \precsim O_j\right)$. Sistemde, eş zamanlı olayların farklı nesneler üzerinde gerçekleştiği varsayımı yapılmaktadır. Bir $O_i$ olayının her bir $p_i$ parametresi ortamdaki bir nesneye ($\mathcal{N}$) karşı düşmektedir $\left(Arg_{1:p_i}(O_i) = N_{i,k},\ 1 \le k \le p_i\right)$.

Bazı nesneler ortamda farklı yönelimlerde bulunabildiğinden sisteme eklenen $\mathcal{Y}$ ise nesnelere ilişkin bu özellikleri (örneğin, *sağa_dönük* ve *sola_dönük*) tutmaktadır. Bu özelliklerin kullanımı, nesnelerin davranışları modellenirken değişen yönelimlere göre davranış değişimini modelde kapsayabilmek için gereklidir.

Nesnelerin birbirleriyle etkileşimlerinin modellenmesi için kullanılan, nesneler arasındaki doğrudan gözlemlenebilen bağlantı ve ilişkileri gösteren insan seviyesinde bir bilgi tabanı ($\mathcal{I}$) eğitim senaryolarında sisteme girdi olarak sağlanmadığında, nesnelere ilişkin uzamsal bilgi ve nesneler üzerinde gerçekleşen olayların zaman bilgileri eğitim senaryolarına dahil edilerek uzamsal, zamansal ya da uzam-zamansal çıkarsama ile nesneler arasındaki ilişkilerin bulunabilmesi üzerine derinlemesine bir analiz yapılarak bir etmenin nesneler arasındaki etkileşimlerini hangi ölçüde öğrenilebileceği incelenmiştir.

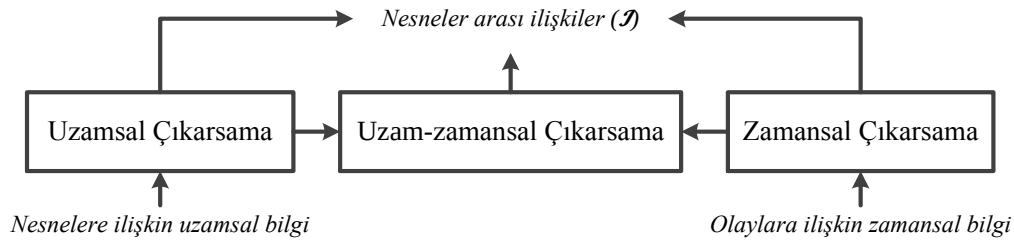
### 5.3 Önerilen Sistemin Genel Yapısı

Nesnelerin işlevlerinin ve birbirleri ile etkileşimlerinin öğrenilmesi problemine çözüm olarak iki aşamalı bir sistem geliştirilmiştir (Şekil 5.1). Sistemin ilk aşamasında gerçekleşen olaylar üzerinden ortamdaki nesneler tiplerine göre gruplanmakta ve her bir nesne grubunun davranışı sonlu durum makineleri (SDM) yardımıyla modellenmektedir. Bu aşamada LOCM (Cresswell, McCluskey, & West, 2009; Cresswell, McCluskey, & West, in press) yönteminin temel aldığı varsayımlardan yararlanılmıştır. Ayrıca, değişen nesne yönelimlerinin nesne davranışlarına etkisinin modellenebilmesi için, ortamdaki nesnelerin ilk durumuna ilişkin yönelim bilgileri sisteme girdi olarak verilerek LOCM yöntemi ile elde edilen sonuçlar iyileştirilmiştir.




**Şekil 5.1:** Nesne işlev ve etkileşimlerinin öğrenildiği sistemin genel yapısı.

İkinci aşamada, farklı nesne gruplarını modellemek için kullanılan sonlu durum makineleri arasında koşullu bağlantılar oluşturularak nesneler arası etkileşimler öğrenilmektedir. Öğrenme aşamasında ortamdaki nesneler arası bağlantı ve ilişkilerden yararlanılmıştır. Bu bağlantılara ilişkin bilgi verilmediğinde ise nesnelerin uzamsal bilgileri ile nesneler üzerinde gerçekleşen olayların zamansal bilgileri üzerinden çıkarsama yapılmaktadır (Şekil 5.2).



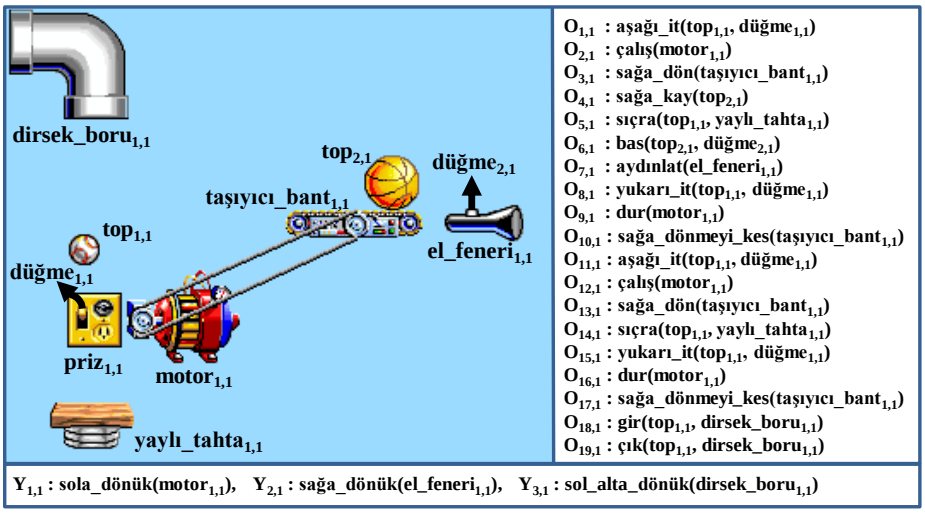

**Şekil 5.2:** Uzamsal ve zamansal bilgi kullanılarak bilgi tabanının oluşturulması.

# 6. NESNE DAVRANIŞLARININ MODELLENMESİ

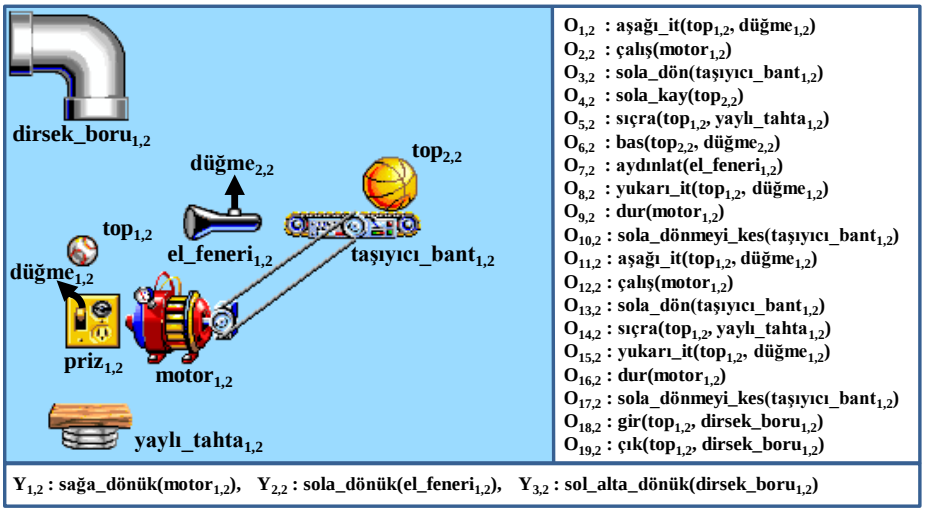
## 6.1 Örnek Eğitim Senaryoları

Geliştirilen sistemin ilk aşamasında nesnelerin işlevleri sonlu durum makineleri ile modellenmektedir. Sistemin bu aşaması tanıtılırken Şekil 6.1 (a) ve Şekil 6.1 (b)'de verilen çeşitli nesneler üzerinde gerçekleşen olaylar için oluşturulan örnek eğitim senaryolarından yararlanılacaktır. Ortamda gerçekleşen olaylar üzerinden senaryonun dinamik akışını modellemek üzere kullanılan metin tabanlı olay dizileri ($O_{1:19,1}$ ve $O_{1:19,2}$) şekillerin sağ tarafında yer almaktadır. Şekillerin altında ise farklı yönelimlerde bulunabilen nesnelerin ilgili senaryoya ilişkin yönelim bilgileri ($Y_{1:3,1}$ ve $Y_{1:3,2}$) verilmiştir. Nesne, olay ve yönelim bilgilerindeki alt indislerde bulunan sayılardan ilki nesne, olay ya da yönelim numarasını, ikincisi ise senaryo numarasını ifade etmektedir. Örneğin, $top_{1,2}$ ile 2. senaryodaki 1. top nesnesi adlandırılmaktadır.

Verilen ilk senaryoda (Şekil 6.1 (a)): yukarıdan bırakılan $top_{1,1}$'in $priz_{1,1}$'i kontrol eden $düğme_{1,1}$'i aşağı itmesiyle $priz_{1,1}$'e bağlı olan $motor_{1,1}$ çalışmaya başlamakta ve kendisine kayışla bağlı olan $taşıyıcı\_bant_{1,1}$'in sağa dönmesini sağlamaktadır. Sağa dönen $taşıyıcı\_bant_{1,1}$'in üzerinde yer alan $top_{2,1}$ sağa kayarak $el\_feneri_{1,1}$'i kontrol eden $düğme_{2,1}$'e basmakta ve böylece $el\_feneri_{1,1}$ yanmaktadır. Bu esnada $yaylı\_tahta_{1,1}$ üzerinden geri dönen $top_{1,1}$, $düğme_{1,1}$'i yukarı iterek $priz_{1,1}$'in elektrik vermeyi kesmesine ve $motor_{1,1}$ ile sürdüğü $taşıyıcı\_bant_{1,1}$'in durmasına neden olmaktadır. Sonra, yer çekiminin etkisiyle tekrar aşağı inen $top_{1,1}$, $düğme_{1,1}$'i aşağı iterek $motor_{1,1}$'in yeniden çalışmasını ve ona bağlı olan $taşıyıcı\_bant_{1,1}$'in sağa dönmesini sağlamaktadır. $top_{1,1}$, son kez $yaylı\_tahta_{1,1}$ üzerinden sektikten sonra $dirsek\_boru_{1,1}$ içinden geçerek aşağı düşmektedir. Şekil 6.1 (b)'deki ikinci senaryonun akışı, ilk senaryoya oldukça benzemektedir. İki senaryo arasındaki fark *motor* ve *el_feneri* nesnelerinin yönelimlerinin değişmiş olmasıdır. Sola dönük olan $motor_{1,1}$ kayışla sürdüğü $taşıyıcı\_bant_{1,1}$'in sağa dönmesini sağlarken, sağa dönük olan $motor_{1,2}$'ye bağlı olan $taşıyıcı\_bant_{1,2}$ ters yönde hareket etmektedir. Ayrıca, $el\_feneri_{1,1}$ ve $el\_feneri_{1,2}$ de birbirinin aksi yönlerde ışık vermektedirler.




**(a)**




**(b)**

**Şekil 6.1:** Nesne davranışlarının modellenmesi için örnek eğitim senaryoları.

## 6.2 LOCM Kullanılarak Sonlu Durum Makinelerinin Oluşturulması

Ortamdaki nesnelerin tiplerine göre gruplanması ve nesne gruplarına ilişkin durum değişimlerinin sonlu durum makineleri yardımıyla modellenmesi için LOCM'nin (Cresswell, McCluskey, & West, 2009; Cresswell, McCluskey, & West, in press) aşağıda belirtilen varsayımlarından yararlanılmıştır.

1) Bir $O_i$ olayının her bir $p_i$ argümanı bir ve yalnız bir nesne grubuna karşı düşmektedir.
2) Her $O_i$ olayı her bir $p_i$ argümanındaki nesneler üzerinde bir durum geçişine neden olur. Bazı geçişlerde nesnenin durumu aynı kalabilir.
3) Bir durum geçişi aynı sonlu durum makinesinde birden fazla kez bulunamaz.

İlk varsayım yardımıyla LOCM ortamdaki nesneleri ayrık kümeler şeklinde gruplara ayırmaktadır. Bu gruplamayla aynı isimli olayların aynı sırada bulunan parametrelerindeki nesneler aynı gruba dahil edilmektedir. Örneğin, *aşağı_it* isimli olay Şekil 6.1'de verilen her iki eğitim senaryosunda da gözlemlenmektedir ($O_{1,1} = aşağı\_it(top_{1,1}, düğme_{1,1})$ ve $O_{1,2} = aşağı\_it(top_{1,2}, düğme_{1,2})$). Bu gözlemden yola çıkılarak *aşağı_it* olayının ilk argümanında geçen $top_{1,1}$ ve $top_{1,2}$ nesneleri birbiriyle aynı gruba alınır ve benzer şekilde ikinci argümanda geçen $düğme_{1,1}$ ve $düğme_{1,2}$ nesneleri de birbiriyle aynı gruba alınır.

İkinci varsayım kullanılarak ortamdaki nesnelerin davranışlarını modellemek üzere her bir nesne grubu için bir sonlu durum makinesi oluşturulmaktadır. Sonlu durum makineleri oluşturma işleminde öncelikle her bir $O_i$ olayının $p_i$ argümanında verilen nesneler için durum geçişleri tanımlanmaktadır. Sonra aynı nesne üzerindeki birbirini takip eden durum geçişleri dikkate alınarak, eşdeğer durum tespiti yapılmakta ve sonlu durum makinelerindeki durumlar indirgenmektedir. Örneğin, Şekil 6.1 (a)'daki $O_{3,1} = sağa\_dön$ ve $O_{10,1} = sağa\_dönmeyi\_kes$ olayları $taşıyıcı\_bant_{1,1}$ nesnesini argüman olarak içermektedir. 2. varsayım yardımıyla öncelikle bu olayların $taşıyıcı\_bant_{1,1}$ üzerindeki durum geçişleri modellenebilir (6.1). Durumlar ifade edilirken ($durum_{i,j}$), indislemede kullanılan sayılardan ilki ($i$) durum numarasını, ikincisi ise ($j$) nesnenin grup numarasını ifade etmektedir. $taşıyıcı\_bant_{1,1}$ nesnesi olay listesindeki farklı gruba ait nesneler arasında dördüncü sırada yer aldığından, bu nesnenin grup numarası olarak 4 atanmıştır.

$$durum_{1,4} \xrightarrow{sağa\_dön_1} durum_{2,4}$$
$$durum_{3,4} \xrightarrow{sağa\_dönmeyi\_kes_1} durum_{4,4}$$

**(6.1)**

Bu olaylar arasında $taşıyıcı\_bant_{1,1}$ nesnesi üzerinde gerçekleşen başka bir olay bulunmadığından, bu nesnenin $O_{3,1} = sağa\_dön$ olayı sonrasındaki durumu ($durum_{2,4}$) ile $O_{10,1} = sağa\_dönmeyi\_kes$ olayı öncesindeki durumunun ($durum_{3,4}$) aynı olduğu sonucuna varılmaktadır (6.2).

$$durum_{1,4} \xrightarrow{sağa\_dön_1} durum_{2,4}$$
$$durum_{2,4} \xrightarrow{sağa\_dönmeyi\_kes_1} durum_{4,4}$$

**(6.2)**

$taşıyıcı\_bant_{1,1}$ nesnesi üzerinde, $O_{10,1} = sağa\_dönmeyi\_kes$ olayı sonrasında ilk gerçekleşen olay $O_{13,1} = sağa\_dön$ olayıdır. Bu bilgi kullanılarak, $durum_{1,4}$ ve $durum_{4,4}$'ün de birbiri ile eşdeğer olduğu sonucuna varılmaktadır (6.3). İfade 6.3 ile, *sağa_dön* ve *sağa_dönmeyi_kes* olaylarının $taşıyıcı\_bant_{1,1}$ nesnesi üzerinde gerçekleştirdiği durum geçişlerinin doğru olarak modellendiği görülmektedir. Benzer işlemler Şekil 6.1 (b)'deki eğitim senaryosunda uygulanarak *sola_dön* ve *sola_dönmeyi_kes* olaylarının *taşıyıcı_bant* üzerindeki etkileri öğrenilmektedir.

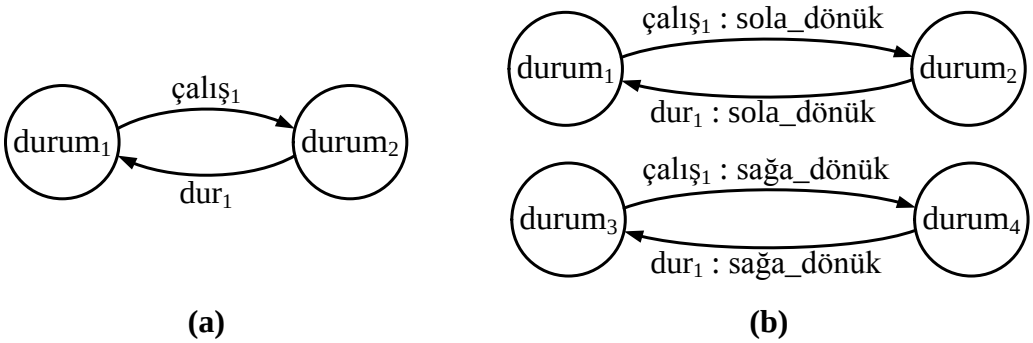
$$durum_{1,4} \xrightarrow{sağa\_dön_1} durum_{2,4}$$
$$durum_{2,4} \xrightarrow{sağa\_dönmeyi\_kes_1} durum_{1,4}$$

**(6.3)**

LOCM yardımıyla üretilen sonlu durum makinelerinde, aynı nesne üzerinde birbirini takip eden durum geçişleri üzerinden durum indirgemesi yapıldığından, üçüncü varsayımda da belirtildiği gibi aynı durum geçişinin bir sonlu durum makinesinde birden fazla kez görülmesine izin verilmemektedir.

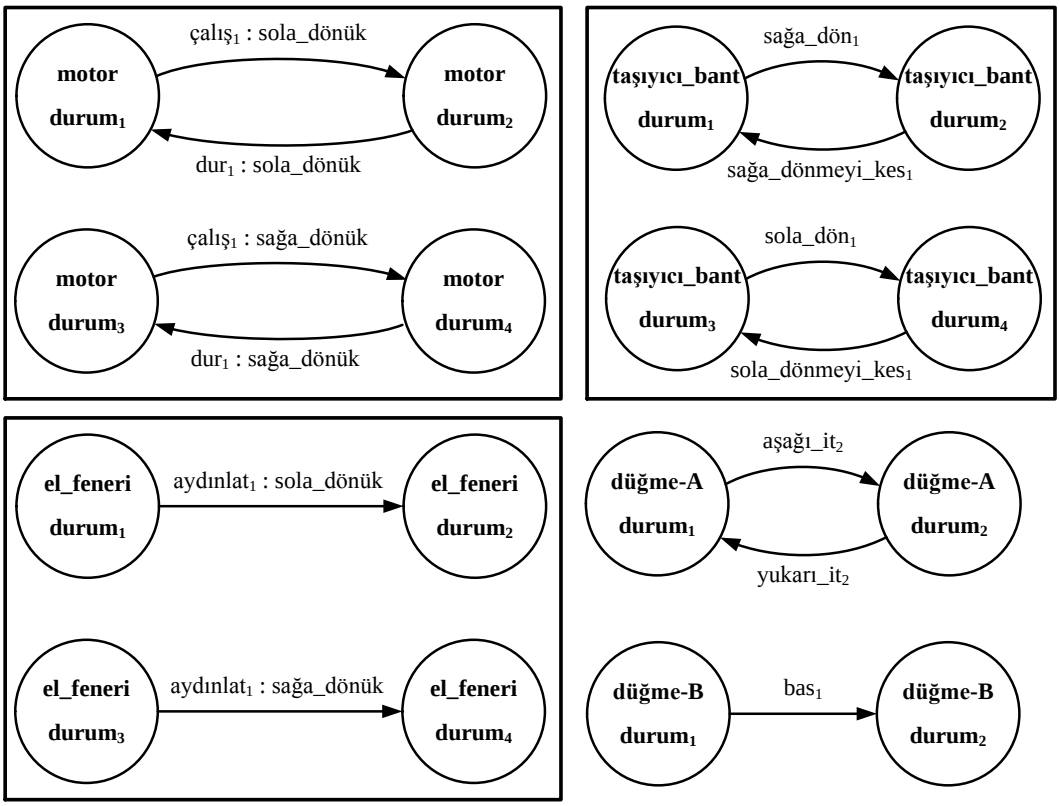
### 6.3 Değişen Nesne Yönelimlerine Göre Modelin Güncellenmesi

LOCM yönteminin varsayımları nesnelerin davranışlarını modellemek üzere iyi bir temel oluşturmaktadır. Fakat üçüncü varsayımın kısıtlaması nedeniyle, Şekil 6.1'deki $motor_{1,1}$ ve $motor_{1,2}$ nesnelerinde olduğu gibi değişen yönelimlere göre aynı olayın farklı etkiler yaratabileceği göz önüne alınmamaktadır. Örnek senaryolardaki motorlar farklı yönlere dönük olduğundan bu motorlara bağlı taşıyıcı bantlar da farklı yönlerde hareket etmektedir. Ancak her iki hareket de aynı isimli $O_{2,1} = O_{2,2} = çalış$ olayı ile ilişkili olduğundan LOCM yöntemi Şekil 6.2 (a)'daki gibi tek bir sonlu durum makinesi üretmektedir. Olay isimleri birbiri ile aynı olduğundan, durum geçişi temelli bir yöntem olan LOCM2 de (Cresswell & Gregory, 2011) LOCM ile aynı sonucu vermektedir. Bu sonlu durum makinesi ile motorun farklı yönlerdeki döndürme etkisi modellenemediğinden, geliştirdiğimiz sistemde bu tip değişken yönelimli nesnelere ilişkin bilgiler ($Y_{1,1} = sola\_dönük(motor_{1,1})$ ve $Y_{1,2} = sağa\_dönük(motor_{1,2})$) sisteme girdi olarak verilerek Şekil 6.2 (b)'de görülebileceği gibi her bir yönelim için ayrı bir sonlu durum makinesi oluşturulması sağlanmıştır.



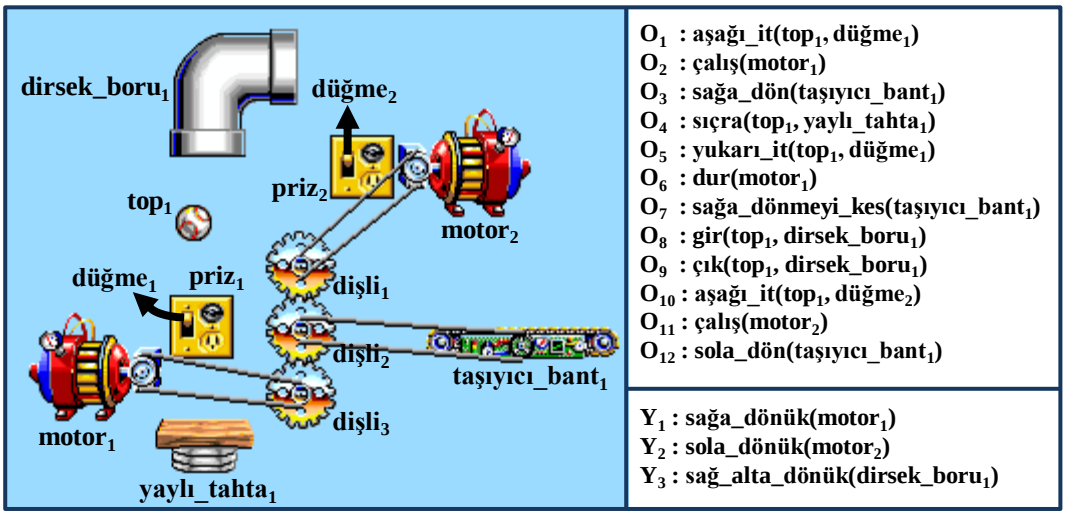

**Şekil 6.2:** *motor* nesne tipi için (a) LOCM ve (b) sistemimizin ürettiği SDMler.

Sistemimizin ilk aşamasında, LOCM yöntemi kullanılarak ve farklı nesne yönelimlerine göre nesne davranışlarının değişebileceği varsayımı altında nesnelerin davranışları (işlevleri) sonlu durum makineleriyle modellenmektedir. Şekil 6.1'deki eğitim senaryoları kullanılarak, bazı nesne gruplarına (*düğme*, *motor*, *taşıyıcı_bant* ve *el_feneri*) ilişkin öğrenilen sonlu durum makineleri Şekil 6.3'te gösterilmektedir. Bu sonlu durum makinelerinde, *taşıyıcı_bant* dışındaki nesnelerin davranışları doğru olarak belirlenebilmiştir. Olaylar üzerinden yapılan gruplama işleminde, *düğme* nesneleri tek bir grupta toplanamadığından *düğme-A* (*priz* düğmesi) ve *düğme-B* (*el_feneri* düğmesi) şeklinde iki ayrı *düğme* grubu bulunmaktadır.

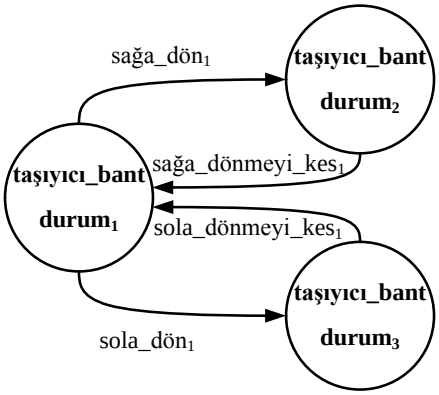



**Şekil 6.3:** Ortamdaki bazı nesneler için eğitim senaryolarından öğrenilen SDMler.

Şekil 6.1'deki eğitim senaryolarında aynı *taşıyıcı_bant* nesnesinin hem sağa hem de sola dönüş yapabildiği gözlemlenmediği için, Şekil 6.3'te *taşıyıcı_bant* için bulunan sonlu durum makinesi eksiktir. *taşıyıcı_bant* nesnesine ilişkin durum geçişlerini tam olarak modelleyebilmek için Şekil 6.4'teki gibi aynı taşıyıcı bandın farklı yönlerde dönebildiğini gösteren bir eğitim senaryosunun Şekil 6.1'deki senaryolara ek olarak kullanılması gerekir. Bu senaryoda araya dişliler eklenerek taşıyıcı bandın önce ilk motorun çalışmasıyla sağa dönmesi, sonra da ikinci motor tarafından sürülerek sola dönmesi sağlanmıştır. Bu senaryodaki olaylarda, $taşıyıcı\_bant_1$ nesnesi üzerinde $O_7 = sağa\_dönmeyi\_kes$ olayını takiben $O_{12} = sola\_dön$ olayının gerçekleştiği gözlemlendiğinden, bu nesnenin $sağa\_dönmeyi\_kes$ olayı sonrasındaki durumu ile $sola\_dön$ olayı öncesindeki durumunun eşdeğer olduğu belirlenmekte ve Şekil 6.5'te verilen sonlu durum makinesi üretilmektedir. Bu sonlu durum makinesi ile *taşıyıcı_bant* nesnesinin davranışı tam olarak modellenebilmektedir.




**Şekil 6.4:** Taşıyıcı bandın farklı yönlerde dönüşünü birlikte gösteren senaryo.




**Şekil 6.5:** *taşıyıcı_bant* nesnesini tam olarak modelleyen SDM.

### 6.4 Nesne Davranışlarını Öğrenme Algoritması

Nesnelerin gruplara ayrılması ve her bir nesne grubu üzerinde gerçekleşen olayların neden olduğu durum geçişlerinin sonlu durum makineleri ile modellenmesi için gerçeklenen sistemin ilk aşamasının genel yapısı Algoritma 6.1'de verilmiştir. Sisteme girdi olarak olayların sıralı dizisi ($\mathcal{O}$) ile farklı yönelimlerde bulunabilen nesnelere ilişkin yönelim bilgisi ($\mathcal{Y}$) verilmektedir. Bu girdiler kullanılarak, öncelikle nesneler gruplanmakta ve nesneler üzerindeki durum geçişleri tanımlanmaktadır. Sonrasında aynı nesne üzerinde birbirini takip eden durum geçişleri üzerinden eşdeğer durum indirgemesi yapılarak, nesne gruplarının davranışlarını modelleyen sonlu durum makineleri elde edilmektedir. Tüm bu işlemlerin sonunda çıktı olarak, sonlu durum makinelerini oluşturmada kullanılan durum geçişleri, nesne grup bilgileri, nesne ve olay kümeleri üretilmektedir.

**Algoritma 6.1:** $nesne\_davranislarini\_modelle(\mathcal{O}, \mathcal{Y})$

**Girdiler:** Olayların sıralı dizisi ($\mathcal{O}$), nesne yönelim bilgisi ($\mathcal{Y}$)
**Çıktılar:** SDM durum geçişleri ($\mathcal{D}$), nesne grupları ($\mathcal{G}$), nesne ve olay kümeleri ($\mathcal{NK}$, $\mathcal{OK}$)

1: $\mathcal{G}, \mathcal{NK}, \mathcal{OK}, \mathcal{D}, grup\_sayisi, \mathcal{GY} \leftarrow gruplari\_belirle(\mathcal{O}, \mathcal{Y})$
2: $\mathcal{D} \leftarrow durum\_gecislerini\_ilklendir(\mathcal{D}, \mathcal{GY}, grup\_sayisi)$
3: $\mathcal{D} \leftarrow durum\_indirgemesi\_yap(\mathcal{D}, \mathcal{Y}, \mathcal{O})$
4: **döndür** $\mathcal{D}, \mathcal{G}, \mathcal{NK}, \mathcal{OK}$

LOCM'nin 1. varsayımından yararlanılarak nesnelerin gruplara ayrılması işlemi Algoritma 6.2'de verilen adımlarla yapılmaktadır. Olaylar listesinden ($\mathcal{O}$) sırayla okunan olayların argümanlarındaki nesneler gruplanırken dört farklı durum söz konusu olabilir. Eğer ilk kez gözlemlenen bir olayın argümanında daha önce gözlemlenmemiş bir nesne verilmişse, bu nesne ile yeni bir grup oluşturulur (satır 10-15). İlk kez gözlemlenen bir olayın argümanında daha önce gözlemlenmiş olan bir nesne verildiğinde, olayın ilgili argümanına ilişkin grup bilgisi olarak nesnenin grup bilgisi kullanılır (satır 16 ve 17). Eğer daha önce gözlemlenmiş bir olayın bir argümanında daha önceden gözlemlenmemiş bir nesne verilmişse, bu nesnenin grup bilgisini belirlemede olayın ilgili argümanının grup bilgisi kullanılır (satır 18-22). Daha önce gözlemlenmiş olan bir olayın argümanında daha önce gözlemlenmiş bir nesne verildiğinde, grup numaraları birbirinden farklı ise gruplar (grup numarası daha küçük olan grup altında) birleştirilir (satır 23-29). Grupların birleştirilmesi işlemi, Algoritma 6.3'teki adımlarla yapılmaktadır.

**Algoritma 6.2:** $gruplari\_belirle(\mathcal{O}, \mathcal{Y})$

**Girdiler:** Olayların sıralı dizisi ($\mathcal{O}$), nesne yönelim bilgisi ($\mathcal{Y}$)
**Çıktılar:** Nesnelere ilişkin grup bilgisi ($\mathcal{G}$), nesneler kümesi ($\mathcal{NK}$), olaylar kümesi ($\mathcal{OK}$), SDMlerin durum geçişleri ($\mathcal{D}$), grup sayısı ($grup\_no$), grup yönelimleri ($\mathcal{GY}$)

1: $grup\_no \leftarrow 0$
2: $\mathcal{OK} \leftarrow \emptyset$ ve $\mathcal{NK} \leftarrow \emptyset$
3: **her** $O_i \in \mathcal{O}$ **için**
4: **eğer** $O_i \notin \mathcal{OK}$ **ise**
5: $yeni\_olay \leftarrow true$
6: $ekle\ O_i \rightarrow \mathcal{OK}$
7: **değilse**
8: $yeni\_olay \leftarrow false$
9: **her** $1 \leq k \leq p_i$ **için**
10: **eğer** $yeni\_olay$ **ve** $O_{i,k} \notin \mathcal{NK}$ **ise**
11: $ekle\ O_{i,k} \rightarrow NK$
12: $\mathcal{G}(O_{i,k}) \leftarrow grup\_no + 1,\ \mathcal{D}(O_i, k, 1) \leftarrow grup\_no + 1$
13: $grup\_no \leftarrow grup\_no + 1$
14: **eğer** $\mathcal{Y}(O_{i,k}) \notin \mathcal{GY}(grup\_no)$ **ise**
15: $ekle\ \mathcal{Y}(O_{i,k}) \rightarrow \mathcal{GY}(grup\_no)$
16: **eğer** $yeni\_olay$ **ve** $O_{i,k} \in \mathcal{NK}$ **ise**
17: $\mathcal{D}(O_i, k, 1) \leftarrow \mathcal{G}(O_{i,k})$
18: **eğer** ! $yeni\_olay$ **ve** $O_{i,k} \notin \mathcal{NK}$ **ise**
19: $ekle\ O_{i,k} \rightarrow \mathcal{NK}$
20: $\mathcal{G}(O_{i,k}) \leftarrow \mathcal{D}(O_i, k, 1)$
21: **eğer** $\mathcal{Y}(O_{i,k}) \notin \mathcal{GY}\left(\mathcal{G}(O_{i,k})\right)$ **ise**
22: $ekle\ \mathcal{Y}(O_{i,k}) \rightarrow \mathcal{GY}\left(\mathcal{G}(O_{i,k})\right)$
23: **eğer** ! $yeni\_olay$ **ve** $O_{i,k} \in \mathcal{NK}$ **ise**
24: **eğer** $\mathcal{G}(O_{i,k}) < \mathcal{D}(O_i, k, 1)$ **ise**
25: $\mathcal{G}, \mathcal{D}, \mathcal{GY} \leftarrow grup\_birlestir\left(\mathcal{G}, \mathcal{D}, \mathcal{GY}, \mathcal{D}(O_i, k, 1), \mathcal{G}(O_{i,k})\right)$
26: $grup\_no \leftarrow grup\_no - 1$
27: **eğer** $\mathcal{G}(O_{i,k}) > \mathcal{D}(O_i, k, 1)$ **ise**
28: $\mathcal{G}, \mathcal{D}, \mathcal{GY} \leftarrow grup\_birlestir\left(\mathcal{G}, \mathcal{D}, \mathcal{GY}, \mathcal{G}(O_{i,k}), \mathcal{D}(O_i, k, 1)\right)$
29: $grup\_no \leftarrow grup\_no - 1$
30: **döndür** $\mathcal{G}, \mathcal{NK}, \mathcal{OK}, \mathcal{D}, grup\_no, \mathcal{GY}$

Gözlemlenen nesnelerin yönelim bilgilerini nesne grupları ile ilişkilendiren grup yönelimleri ($\mathcal{GY}$) listesi de gruplarla beraber oluşturulmaktadır. Nesne kümesine ($\mathcal{NK}$) her yeni nesne eklendiğinde, eklenen nesnenin yönelim bilgisi de o nesnenin dahil olduğu grubun yönelim bilgilerinin tutulduğu listeye eklenmektedir (satır 15 ve satır 22). Algoritma 6.3'te verilen adımlarla grup birleştirmesi işlemi yapılırken, birleştirilen gruplardaki nesnelere ilişkin gözlemlenen farklı yönelim bilgilerini tutan listeler de birleştirilmektedir (satır 1-7).

**Algoritma 6.3:** $grup\_birlestir(\mathcal{G}, \mathcal{D}, \mathcal{GY}, eski\_grup\_no, yeni\_grup\_no)$

**Girdiler:** Nesnelere ilişkin grup bilgisi ($\mathcal{G}$), SDMlerin durum geçişleri ($\mathcal{D}$), grup yönelimleri ($\mathcal{GY}$), değiştirilecek grup numarası ($eski\_grup\_no$), atanacak grup numarası ($yeni\_grup\_no$)
**Çıktılar:** Nesnelere ilişkin grup bilgisi ($\mathcal{G}$), SDMlerin durum geçişleri ($\mathcal{D}$), grup yönelimleri ($\mathcal{GY}$)

1: **her** $Y_i \in \mathcal{GY}(eski\_grup\_no)$ **için**
2: **eğer** $Y_i \notin \mathcal{GY}(yeni\_grup\_no)$ **ise**
3: $ekle\ Y_i \rightarrow \mathcal{GY}(yeni\_grup\_no)$
4: $grup\_sayisi \leftarrow boyut(\mathcal{GY})$
5: **her** $eski\_grup\_no < G_i \leq grup\_sayisi$ **için**
6: $\mathcal{GY}(G_i - 1) \leftarrow \mathcal{GY}(G_i)$
7: $sil\ \mathcal{GY}(G_i)$
8: **her** $N_i \in \mathcal{G}$ **için**
9: **eğer** $\mathcal{G}(N_i) = eski\_grup\_no$ **ise**
10: $\mathcal{G}(N_i) \leftarrow yeni\_grup\_no$
11: **eğer** $\mathcal{G}(N_i) > eski\_grup\_no$ **ise**
12: $\mathcal{G}(N_i) \leftarrow \mathcal{G}(N_i) - 1$
13: **her** $O_i \in \mathcal{D}$ **için**
14: **her** $1 \leq k \leq p_i$ **için**
15: **eğer** $\mathcal{D}(O_i, k, 1) = eski\_grup\_no$ **ise**
16: $\mathcal{D}(O_i, k, 1) \leftarrow yeni\_grup\_no$
17: **eğer** $\mathcal{D}(O_i, k, 1) > eski\_grup\_no$ **ise**
18: $\mathcal{D}(O_i, k, 1) \leftarrow \mathcal{D}(O_i, k, 1) - 1$
19: **döndür** $\mathcal{G}, \mathcal{D}, \mathcal{GY}$

Gözlemlenen olaylar üzerinden sonlu durum makineleri oluşturulurken, önce Algoritma 6.4'ün adımları uygulanarak her bir farklı olayın her bir argümanı için durum geçişleri tanımlanmaktadır. Ortamda birden fazla yönelimde bulunabilen nesnelerin her farklı yönelimi için ayrı durum geçişleri oluşturulmaktadır (satır 6-8).

**Algoritma 6.4:** $durum\_gecislerini\_ilklendir(\mathcal{D}, \mathcal{GY}, grup\_sayisi)$

**Girdiler:** SDMlerin durum geçişleri ($\mathcal{D}$), grup yönelimleri ($\mathcal{GY}$), grup sayısı ($grup\_sayisi$)
**Çıktılar:** SDMlerin durum geçişleri ($\mathcal{D}$)

1: **her** $1 \leq i \leq grup\_sayisi$ **için**
2: $durum\_no(i) \leftarrow 0$
3: **her** $O_i \in \mathcal{D}$ **için**
4: **her** $1 \leq k \leq p_i$ **için**
5: $grup\_no \leftarrow \mathcal{D}(O_i, k, 1)$
6: **her** $Y_j \in \mathcal{GY}(grup\_no)$ **için**
7: $\mathcal{D}(O_i, k, 2, Y_j) \leftarrow (durum\_no(grup\_no) + 1, durum\_no(grup\_no) + 2)$
8: $durum\_no(grup\_no) \leftarrow durum\_no(grup\_no) + 2$
9: **döndür** $\mathcal{D}$

Durum geçişleri ilklendirildikten sonra, aynı nesne üzerinde birbirini takip eden olayların neden olduğu durum geçişleri üzerinden durum indirgemesi yapılırken uygulanan adımlar Algoritma 6.5'te görülmektedir. Tüm nesneler için son gözlemlendiği olay ve argüman sırası bilgisi tutularak (satır 16), aynı nesne üzerinde birbirini takip eden durum geçişlerinden ilkinin son durumu ile ikincisinin ilk durumu daha küçük numara değerine sahip olan durum altında birleştirilmektedir (satır 9-15).

**Algoritma 6.5:** $durum\_indirgemesi\_yap(\mathcal{D}, \mathcal{Y}, \mathcal{O})$

**Girdiler:** SDMlerin durum geçişleri ($\mathcal{D}$), nesne yönelim bilgisi ($\mathcal{Y}$), olayların sıralı dizisi ($\mathcal{O}$)
**Çıktılar:** SDMlerin durum geçişleri ($\mathcal{D}$)

1: $son\_gozlem \leftarrow \emptyset$
2: **her** $O_i \in \mathcal{O}$ **için**
3: **her** $1 \leq k \leq p_i$ **için**
4: **eğer** $O_{i,k} \in son\_gozlem$ **ise**
5: $sg\_i \leftarrow son\_gozlem(O_{i,k}, 1)$
6: $sg\_k \leftarrow son\_gozlem(O_{i,k}, 2)$
7: $grup\_no \leftarrow \mathcal{D}(O_i, k, 1)$
8: $yon \leftarrow \mathcal{Y}(O_{i,k})$
9: **eğer** $\mathcal{D}(O_i, k, 2, \mathcal{Y}(O_{i,k}), 1) > \mathcal{D}(O_{sg\_i}, sg\_k, 2, \mathcal{Y}(O_{i,k}), 2)$ **ise**
10: $eski\_durum\_no \leftarrow \mathcal{D}(O_i, k, 2, \mathcal{Y}(O_{i,k}), 1)$
11: $yeni\_durum\_no \leftarrow \mathcal{D}(O_{sg\_i}, sg\_k, 2, \mathcal{Y}(O_{i,k}), 2)$
12: **eğer** $\mathcal{D}(O_i, k, 2, \mathcal{Y}(O_{i,k}), 1) < \mathcal{D}(O_{sg\_i}, sg\_k, 2, \mathcal{Y}(O_{i,k}), 2)$ **ise**
13: $eski\_durum\_no \leftarrow \mathcal{D}(O_{sg\_i}, sg\_k, 2, \mathcal{Y}(O_{i,k}), 2)$
14: $yeni\_durum\_no \leftarrow \mathcal{D}(O_i, k, 2, \mathcal{Y}(O_{i,k}), 1)$
15: $\mathcal{D} \leftarrow durum\_birleştir(\mathcal{D}, grup\_no, yon, eski\_durum\_no, yeni\_durum\_no)$
16: $son\_gozlem(O_{i,k}) \leftarrow (i, k)$
17: **döndür** $\mathcal{D}$

Durumların indirgemesi işleminde Algoritma 6.6'daki adımlar uygulanarak iki durum küçük numaralı olan durumun altında birleştirilmekte ve diğer durum numaraları birbirini takip edecek şekilde yeniden düzenlenmektedir. Durum birleştirmesi yapılacak olan sonlu durum makinesindeki tüm durum geçişlerinde, birleştirilecek durumlardan büyük numaraya sahip olana diğer durumun numarası atanmaktadır (satır 4-7). Ayrıca, durum numaralarının arada atlamalar olmaksızın birbirini takip etmeleri için, birleştirme esnasında silinen durum numarasından (birleştirilen iki durumdan büyük numaraya sahip olanın numara değeri) daha büyük numara değerine sahip olan durumların numaraları bir azaltılmaktadır (satır 9-13).

**Algoritma 6.6:** $durum\_birlestir(\mathcal{D}, grup\_no, yon, eski\_durum\_no, yeni\_durum\_no)$

**Girdiler:** SDMlerin durum geçişleri ($\mathcal{D}$), grup bilgisi ($grup\_no$), nesne yönelim bilgisi ($yon$), değiştirilecek durum no ($eski\_durum\_no$), atanacak durum no ($yeni\_durum\_no$)
**Çıktılar:** SDMlerin durum geçişleri ($\mathcal{D}$)

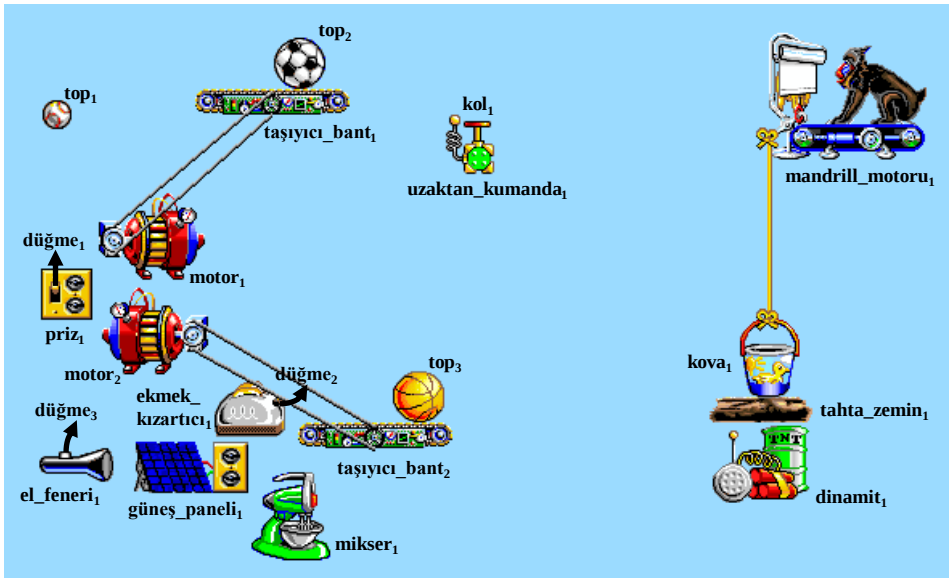
**her** $O_i \in \mathcal{D}$ **için**
  **her** $1 \leq k \leq p_i$ **için**
    **eğer** $\mathcal{D}(O_i, k, 1) = grup\_no$ **ise**
      **eğer** $\mathcal{D}(O_i, k, 2, yon, 1) = eski\_durum\_no$ **ise**
        $\mathcal{D}(O_i, k, 2, yon, 1) \leftarrow yeni\_durum\_no$
      **eğer** $\mathcal{D}(O_i, k, 2, yon, 2) = eski\_durum\_no$ **ise**
        $\mathcal{D}(O_i, k, 2, yon, 2) \leftarrow yeni\_durum\_no$
      $y \leftarrow boyut\big(\mathcal{D}(O_i, k, 2)\big)$
      **her** $1 \leq j \leq y$ **için**
        **eğer** $\mathcal{D}(O_i, k, 2, j, 1) > eski\_durum\_no$ **ise**
          $\mathcal{D}(O_i, k, 2, j, 1) \leftarrow \mathcal{D}(O_i, k, 2, j, 1) - 1$
        **eğer** $\mathcal{D}(O_i, k, 2, j, 2) > eski\_durum\_no$ **ise**
          $\mathcal{D}(O_i, k, 2, j, 2) \leftarrow \mathcal{D}(O_i, k, 2, j, 2) - 1$
**döndür** $\mathcal{D}$

# 7. NESNELER ARASI ETKİLEŞİMLERİN BELİRLENMESİ

Önerilen sistemin ikinci aşamasında, ortamdaki farklı nesne grupları arasındaki ilişkilerden yararlanılarak, gözlemlenen olaylar üzerinden nesneler arasındaki etkileşimler öğrenilmektedir. Bu amaçla, nesneler arasındaki bağlantıları modellemek üzere ortamda doğrudan gözlenebilen ilişkileri içeren insan seviyesinde bir bilgi tabanı, uzamsal bilgi, zamansal bilgi ve uzam-zamansal bilgi şeklinde dört farklı girdi tipinin öğrenme sonucuna etkisi analiz edilmiştir.

## 7.1 Örnek Eğitim Senaryosu

Sistemin, nesneler arasındaki etkileşimlerin öğrenilmesine yönelik olan ikinci aşaması tanıtılırken faydalanılacak olan örnek eğitim senaryosunun ilk durumu Şekil 7.1'de görülmektedir. Bu senaryonun dinamik akışını metin tabanında modellemek üzere, Şekil 7.2 (a)'da verilen, nesneler üzerindeki olayların sıralı dizisi ($O_{1:17}$) kullanılmaktadır. Şekil 7.2 (b)'de görülen $Y_{1:5}$ ise ortamda farklı yönelimlerde bulunabilen *motor*, *mandrill_motoru*, *el_feneri* ve *ekmek_kızartıcı* nesnelerinin bu eğitim senaryosunda gözlemlenen yönelim bilgilerini taşımaktadır.




**Şekil 7.1:** Çeşitli nesneler arasındaki etkileşimleri barındıran örnek senaryo.

Verilen örnek senaryoda, yukarıdan bırakılan $top_1$'in $priz_1$'i kontrol eden $düğme_1$'i aşağı itmesiyle iki zincirleme tepki başlamaktadır ve ortamdaki nesneler arasında $O_{1:17}$ dizisinde belirtilen olaylar gerçekleşmektedir.

| | |
|---|---|
| $O_1$ : *aşağı_it*(*top*$_1$, *düğme*$_1$) | $Y_1$ : *sola_dönük*(*motor*$_1$) |
| $O_2$ : *çalış*(*motor*$_1$) | $Y_2$ : *sağa_dönük*(*motor*$_2$) |
| $O_3$ : *çalış*(*motor*$_2$) | $Y_3$ : *sola_dönük*(*mandrill_motoru*$_1$) |
| $O_4$ : *sağa_dön*(*taşıyıcı_bant*$_1$) | $Y_4$ : *sağa_dönük*(*el_feneri*$_1$) |
| $O_5$ : *sola_dön*(*taşıyıcı_bant*$_2$) | $Y_5$ : *sağa_dönük*(*ekmek_kızartıcı*$_1$) |
| $O_6$ : *sağa_kay*(*top*$_2$) | |
| $O_7$ : *sola_kay*(*top*$_3$) | |
| $O_8$ : *bas*(*top*$_1$, *düğme*$_3$) | |
| $O_9$ : *aydınlat*(*el_feneri*$_1$) | |
| $O_{10}$ : *karıştır*(*mikser*$_1$) | |
| $O_{11}$ : *bas*(*top*$_3$, *düğme*$_2$) | |
| $O_{12}$ : *çarp*(*top*$_2$, *kol*$_1$) | |
| $O_{13}$ : *etkinleş*(*uzaktan_kumanda*$_1$) | |
| $O_{14}$ : *ekmek_kızart*(*ekmek_kızartıcı*$_1$) | |
| $O_{15}$ : *patla*(*dinamit*$_1$) | |
| $O_{16}$ : *alçal*(*kova*$_1$) | |
| $O_{17}$ : *koşmaya_başla*(*mandrill_motoru*$_1$) | |
| **(a)** | **(b)** |

**Şekil 7.2:** Örnek eğitim senaryosundaki (a) olaylar ve (b) nesne yönelimleri.

Bu örnek eğitim senaryosunda, $motor_1$, $motor_2$, $mikser_1$ ve $ekmek\_kızartıcı_1$ elektrikle çalışan nesnelerdir. $motor_1$ ve $motor_2$, $düğme_1$ ile kontrol edilen $priz_1$'e takılı durumdadır ve $ekmek\_kızartıcı_1$ ile $mikser_1$ ışıktan elektrik üretebilen $güneş\_paneli_1$'e bağlıdır. $el\_feneri_1$ nesnesi $güneş\_paneli_1$'e doğrultulmuştur. $ekmek\_kızartıcı_1$'in ekmek kızartabilmesi için üzerindeki $düğme_2$'ye basılmalıdır. sola dönük olan $motor_1$ kayış bağlantısıyla $taşıyıcı\_bant_1$ nesnesini sürmektedir ve benzer şekilde sağa dönük olan $motor_2$ de $taşıyıcı\_bant_2$'ye kayışla bağlıdır. $mandrill\_motoru_1$'in önündeki perdeyi kontrol eden kola iple bağlı olan $kova_1$, $tahta\_zemin_1$ üzerinde durmaktadır. $uzaktan\_kumanda_1$ üzerindeki $kol_1$ ile kontrol edilen $dinamit_1$ patladığında üzerindeki $tahta\_zemin_1$ yok olmaktadır. Muzlarla arasındaki perde kalktığında $mandrill\_motoru_1$'i süren mandrill koşmaya başlamaktadır.

## 7.2 Bilgi Tabanlı Yaklaşım

Farklı nesne tiplerini modellemek üzere ilk aşamada oluşturulan sonlu durum makineleri arasındaki koşullu etkileşimleri sistemin öğrenebilmesi için, ilk olarak ortamda doğrudan gözlemlenebilen ilişkileri içeren bir bilgi tabanı oluşturulmuştur. Şekil 7.1'deki eğitim senaryosuna ilişkin oluşturulan bilgi tabanındaki bağlantı yapıları Çizelge 7.1'de ilgili örnekler ve bağlantı yönleriyle birlikte verilmiştir.

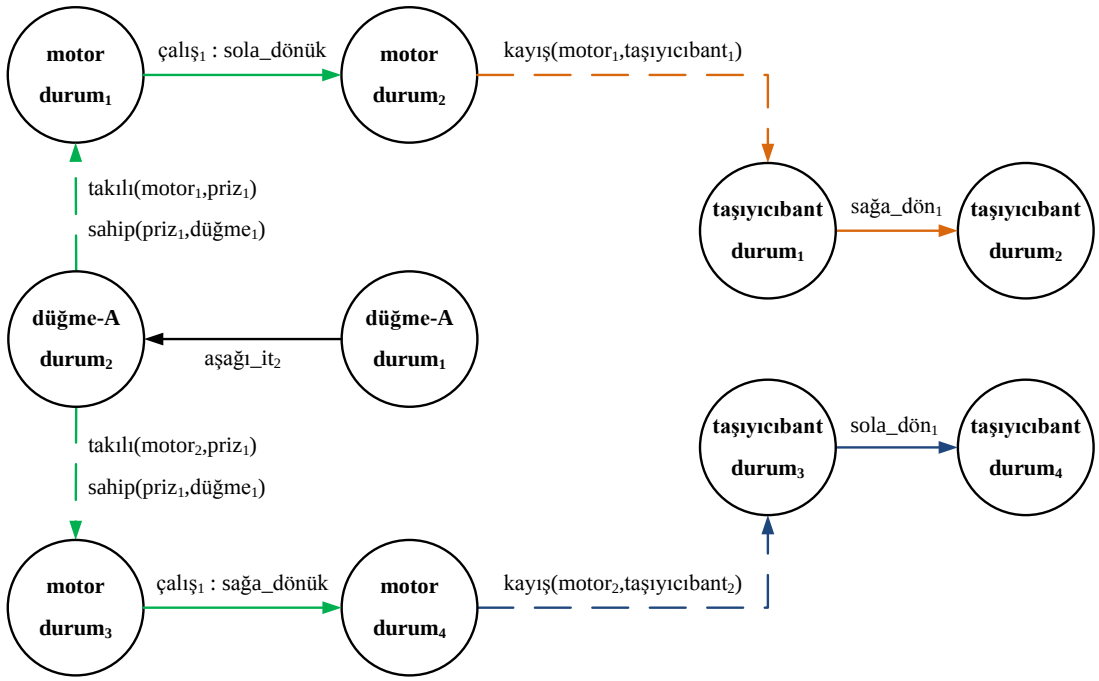
**Çizelge 7.1:** Nesneler arasındaki ilişki tipleri.

| **İlişki ifadesi** | **Şekil 7.1'den örnek** | **Etki yönü** |
|---|---|---|
| *takılı*($n_1$,$n_2$) | *takılı*($motor_1$,$priz_1$)<br>*takılı*($motor_2$,$priz_1$)<br>*takılı*($mikser_1$,$güneş\_paneli_1$)<br>*takılı*($ekmek\_kızartıcı_1$,$güneş\_paneli_1$) | $n_2 \rightarrow n_1$ |
| *üstünde*($n_1$,$n_2$) | *üstünde*($top_2$,$taşıyıcı\_bant_1$)<br>*üstünde*($top_3$,$taşıyıcı\_bant_2$)<br>*üstünde*($kova_1$,$tahta\_zemin_1$) | |
| *kayış*($n_1$,$n_2$) | *kayış*($motor_1$,$taşıyıcı\_bant_1$)<br>*kayış*($motor_2$,$taşıyıcı\_bant_2$) | $n_1 \rightarrow n_2$ |
| *dönük*($n_1$,$n_2$) | *dönük* ($el\_feneri_1$,$güneş\_paneli_1$) | |
| *sahip*($n_1$,$n_2$) | *sahip*($priz_1$,$düğme_1$)<br>*sahip*($ekmek\_kızartıcı_1$,$düğme_2$)<br>*sahip*($el\_feneri_1$,$düğme_3$)<br>*sahip*($uzaktan\_kumanda_1$,$kol_1$) | $n_1 \leftrightarrow n_2$ |
| *ip*($n_1$,$n_2$) | *ip*($mandrill\_motoru_1$,$kova_1$) | |
| *yakın*($n_1$,$n_2$) | *yakın*($dinamit_1$,$tahta\_zemin_1$) | |

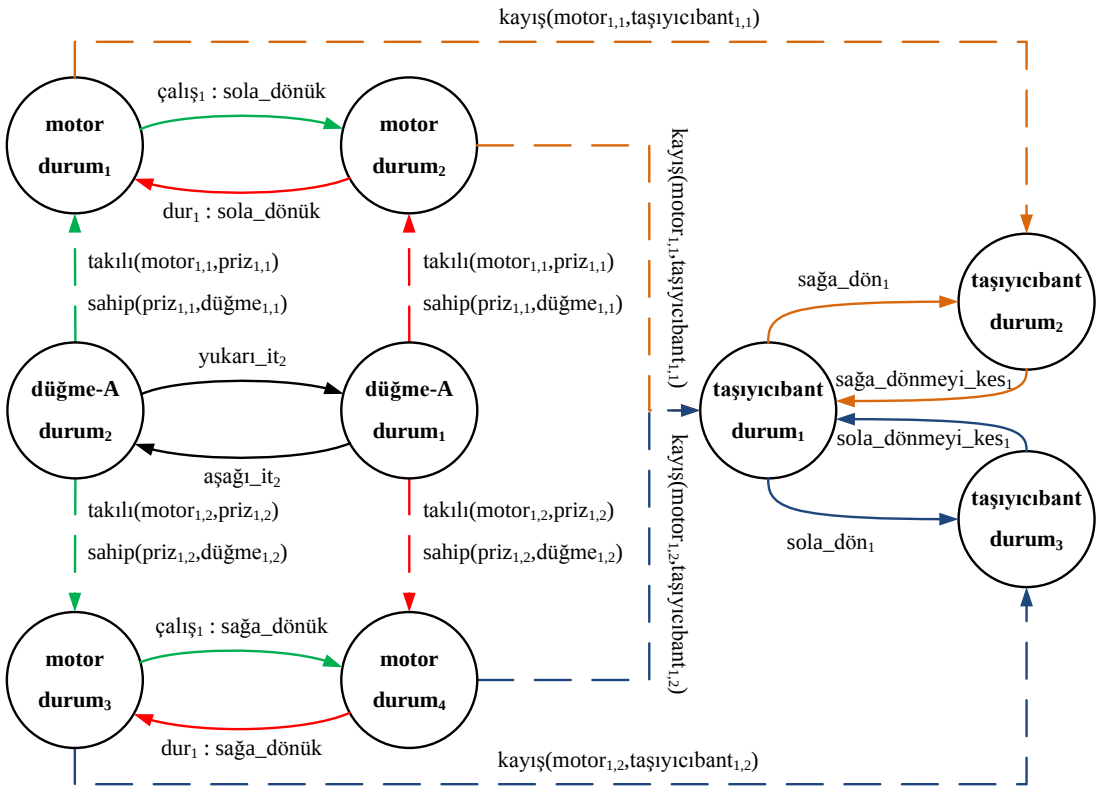
Bu olgusal ifadeler sağlandığında sistem, ileri yönde çıkarsamalar zincirine dayalı bir yapı kullanarak nesneler arası bağlantı kümelerini oluşturmaktadır. Örneğin, Şekil 7.1'deki $ekmek\_kızartıcı_1$ nesnesi için Çizelge 7.1'de iki adet bağlantı mevcuttur: $sahip(ekmek\_kızartıcı_1, düğme_2)$ ve $takılı(ekmek\_kızartıcı_1, güneş\_paneli_1)$. Bu bağlantılar üzerinden, ileri yönde çıkarsama sistemi $\{[düğme_2], [güneş\_paneli_1]\}$ şeklinde iki ilişki alt kümesi oluşturur. Bağlantılar arasındaki geçişlilik özelliğinden faydalanılarak, bu bağlantılar üzerinden erişilen $dönük(el\_feneri_1, güneş\_paneli_1)$ ve $sahip(el\_feneri_1, düğme_3)$ bağlantıları göz önüne alındığında ilişki alt kümeleri $\{[düğme_2], [güneş\_paneli_1, el\_feneri_1, düğme_3]\}$ halini alır. Verilen olay listesinde bulunmayan $güneş\_paneli_1$ nesnesi bu ilişki alt kümelerinden çıkarıldığında, $ekmek\_kızartıcı_1$ nesnesinin ilişkileri $\{[düğme_2], [el\_feneri_1, düğme_3]\}$ kümesi şeklinde son halini alır. Bu çıkarsama diğer nesneler için de benzer şekilde yapılır.

İşlem sonunda elde edilen bağlantı kümeleri, gözlemlenen olaylar yoluyla nesneler arası etkileşimleri öğrenme sürecinde kullanılır. Sistemde her bir bağlantı altkümesi üzerindeki en son gözlenen olay etkisinin kullanımına dayanan sezgisel bir yöntem ile ortamdaki nesnelerin hangi koşullarda ve hangi olaylar üzerinden birbirlerini etkiledikleri belirlenir. Bu yaklaşımla *ekmek_kızart* olayının önkoşullarını belirlemek için $[düğme_2]$ ve $[el\_feneri_1, düğme_3]$ bağlantı alt kümeleri üzerinde etkili olan en son olaylara bakılır. $O_{14} = ekmek\_kızart$ olayından önce $düğme_2$ nesnesini argüman olarak alan en son gözlemlenen olay $O_{11} = bas(top_3, düğme_2)$ olayıdır. $[el\_feneri_1, düğme_3]$ bağlantı alt kümesindeki nesneler üzerinde en son gözlemlenen olay ise $O_9 = aydınlat(el\_feneri_1)$ olayıdır. Bu bilgilerin ışığında *ekmek_kızart* olayının önkoşulları $sahip(ekmek\_kızartıcı_1, düğme_2)$ ilişkisi altında $düğme_2$ üzerinde $bas$ olayının gerçekleşmesi ile $takılı(ekmek\_kızartıcı_1, güneş\_paneli_1)$ ve $dönük(el\_feneri_1, güneş\_paneli_1)$ ilişkileri sağlandığında $el\_feneri_1$ üzerinde *aydınlat* olayının gerçekleşmesi olarak belirlenir.

Örnek eğitim senaryosundaki (Şekil 7.1) $düğme_1$, $motor_1$, $motor_2$, $taşıyıcı\_bant_1$ ve $taşıyıcı\_bant_2$ nesneleri için sadece bu senaryoyu kullanarak bilgi tabanlı yaklaşım ile bulunan etkileşimler Şekil 7.3'te verilmiştir. Düz çizgilerle verilen bağlantılar, sistemin ilk aşamasında oluşturulan sonlu durum makinelerine aittir. Kesikli çizgilerle belirtilen bağlantılar ise, bilgi tabanlı yaklaşım ile farklı nesne grupları arasında öğrenilen koşullu etkileşimleri göstermektedir. Şekilde, bu örnek eğitim senaryosundaki *motor*, *taşıyıcı_bant* ve *düğme* nesnelerinin davranışlarının ve aralarındaki etkileşimlerin (prizin düğmesi aşağı itildiğinde o prize takılı olan motorun çalışması, sola dönük olan motorun kendisine kayışla bağlı olan taşıyıcı bandı sağa döndürmesi ve sağa dönük olan motorun kendisine kayışla bağlı olan taşıyıcı bandı sola döndürmesi) doğru olarak modellendiği görülmektedir. Bununla birlikte, bu eğitim senaryosunda prizin düğmesinin yukarı itilmesi ile elektrik vermeyi kesmesi ve ona bağlı olan motor ile bu motorun sürdüğü taşıyıcı bandın durması gibi etkileşimler bulunmamaktadır. *motor*, *taşıyıcı_bant* ve *düğme* nesnelerinin davranışları ile etkileşimlerinin tam olarak öğrenilebilmesi için aralarındaki çeşitli ilişkileri barındıran eğitim senaryolarının sisteme sağlanması gerekir. Şekil 6.1 ve Şekil 6.4'teki eğitim senaryoları sisteme girdi olarak verildiğinde, bu nesneler arasındaki etkileşimler tam olarak modellenebilmektedir (Şekil 7.4).




**Şekil 7.3:** Şekil 7.1'deki senaryoda bilgi tabanlı yaklaşımın sonuçları.




**Şekil 7.4:** Şekil 6.1 ve 6.4'teki senaryolarda bilgi tabanlı yaklaşımın sonuçları.

### 7.3 Uzamsal Çıkarsama

Nesneler arası bağlantıları içeren bir bilgi tabanı mevcut olmadığında, sistemin sahip olduğu tek bilgi olayların sıralı dizisi olduğundan nesne grupları arasındaki etkileşimlerin öğrenilmesi mümkün olamamaktadır. The Incredible Machine oyunculara eğitim senaryolarındaki nesneler arasındaki görsel ilişkileri inceleyerek yapıyı öğrenme imkanı tanır. Bu durumda, benzer şekilde bir bilgisayar sisteminin de çıkarsama yapabilmesi için ortamdan gerekli bilgileri alabilmesi gerekir. Bu amaçla çeşitli bilgisayarla görü tekniklerinden faydalanılması mümkündür. Bu aşamada, nesneler arasındaki ilişkileri modelleyen insan seviyesinde bir bilgi tabanı yerine nesnelerin uzamsal bilgileri sisteme girdi olarak verilerek uzamsal bir yaklaşımla nesneler arası ilişkilerin belirlenmesi analiz edilmiştir. Nesnelere ilişkin uzamsal bilginin şablon eşleştirmeye (Brunelli, 2009) dayalı bir yöntemle kolayca elde edilebilecek bir temsil olan kapsayan en küçük dikdörtgen (MBR) temsili (Wood, 2008) ile verildiği varsayılmıştır. Bu bilgiden yararlanıldığında nesneler arasında üç tip ilişki (*yakın*, *teğet* ve *sahip*) aşağıdaki çıkarım ile kurulabilir:

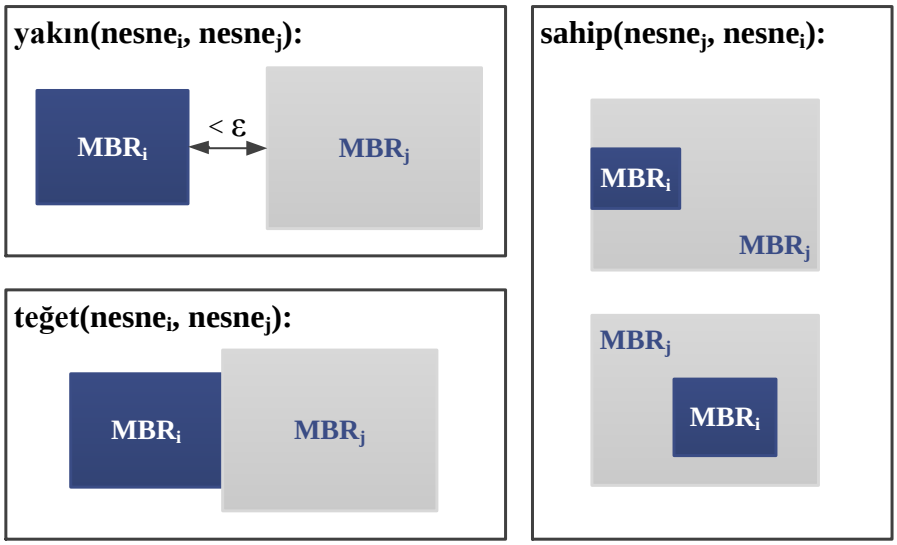
$$0 < uzaklık\left(MBR_i, MBR_j\right) < \varepsilon \Rightarrow yakın\left(nesne_i, nesne_j\right)$$

$$uzaklık\left(MBR_i, MBR_j\right) = 0 \Rightarrow teğet\left(nesne_i, nesne_j\right)$$

$$MBR_i \subset MBR_j \Rightarrow sahip\left(nesne_j, nesne_i\right)$$

**(7.1)**

Bu çıkarımlarda, $MBR_i$, $nesne_i$'nin en küçük kapsayan dikdörtgenini ve $MBR_j$, $nesne_j$'nin en küçük kapsayan dikdörtgenini ifade etmektedir. *teğet* ve *sahip* ilişkileri nesneler arası topolojik duruma göre belirlenen niteliksel ilişkilerdir ve bu ilişkiler oluşturulurken *RCC8* modeli (Randell, Cui, & Cohn, 1992) göz önüne alınmıştır. *yakın* ilişkisi ise nesneler arası uzaklığın belirli bir $\varepsilon$ eşik değerinin altında olmasına dayanan niceliksel bir ilişkidir. Bu ilişkilerin geometrik gösterimi Şekil 7.5'te verilmiştir. Örnek olarak, Şekil 7.1'deki eğitim senaryosuna ilişkin uzamsal çıkarsama ile elde edilen bilgi tabanındaki bağlantılar Çizelge 7.2'de verilmiştir. Bu çizelgedeki *yakın* ilişkisine sahip nesne çiftleri belirlenirken, uzaklık için eşik değer ($\varepsilon$) 15 piksel olarak alınmıştır. Uzamsal çıkarsama ile Çizelge 7.1'deki bilgi tabanlı yaklaşım ifadelerinden *sahip* ilişkisinin ve yakın mesafede bulunan nesneler arasındaki *takılı* ve *üstünde* gibi ilişkilerin algılanabildiği görülmektedir.




**Şekil 7.5:** Uzamsal ilişkilerin geometrik temsili.

**Çizelge 7.2:** Şekil 7.1'deki nesneler arasındaki uzamsal ilişkiler.

| **İlişki ifadesi** | **Şekil 7.1'den örnek** | **Etki yönü** |
|---|---|---|
| *sahip*($n_1$,$n_2$) | *sahip*(priz$_1$,düğme$_1$)<br>*sahip*(ekmek_kızartıcı$_1$,düğme$_2$)<br>*sahip*(el_feneri$_1$,düğme$_3$)<br>*sahip*(uzaktan_kumanda$_1$,kol$_1$) | $n_1 \leftrightarrow n_2$ |
| *teğet*($n_1$,$n_2$) | *teğet*(top$_2$,taşıyıcı_bant$_1$)<br>*teğet*(top$_3$,taşıyıcı_bant$_2$)<br>*teğet*(kova$_1$,tahta_zemin$_1$) | |
| *yakın*($n_1$,$n_2$) | *yakın*(dinamit$_1$,tahta_zemin$_1$)<br>*yakın*(motor$_1$,priz$_1$)<br>*yakın*(motor$_2$,priz$_1$)<br>*yakın*(motor$_1$,motor$_2$)<br>*yakın*(mikser$_1$,güneş_paneli$_1$)<br>*yakın*(ekmek_kızartıcı$_1$,güneş_paneli$_1$)<br>*yakın*(ekmek_kızartıcı$_1$,taşıyıcı_bant$_2$)<br>*yakın*(el_feneri$_1$,güneş_paneli$_1$) | |

Topolojik ilişkilerin yanı sıra ana yönler modelinden (Frank, 1991) faydalanılarak elde edilen nesneler arası bağıl konum bilgisini veren yönsel ilişkiler de çıkarsamada kullanılmıştır. Bazı nesneler arasındaki etkileşimin gerçekleşmesinde nesnelerin birbirlerine göre konumları etkilidir (örneğin, bilgi tabanlı yaklaşımdaki $üstünde(top_2, taşıyıcı\_bant_1)$ ilişkisi), bazı etkileşimlerde ise bunun bir önemi yoktur (örneğin, bilgi tabanlı yaklaşımdaki $takılı(motor_1, priz_1)$ ilişkisi). Bu yaklaşımla, farklı durumları barındıran eğitim senaryoları kullanıldığında bağıl konumların nesneler arası etkileşimde katkısının olup olmadığı belirlenebilir.

### 7.3.1 Uzamsal çıkarsama algoritması

Nesneler arasındaki uzamsal ilişkilerin belirlenmesine ilişkin adımlar, Algoritma 7.1'de verilmiştir. Uzamsal çıkarsama yapılabilmesi için sisteme girdi olarak ortamdaki tüm nesnelerin en küçük kapsayan dikdörtgenlerini (MBR) tutan bir dizi verilmektedir. Her bir MBR'nin ilk elemanı nesne adını, 2. ve 3. sıradaki elemanları sol üst köşe noktanın $x$ ve $y$ piksel değerlerini, 4. sıradaki elemanı dikdörtgenin yatay kenar uzunluğunu $(a)$ ve 5. sıradaki elemanı dikdörtgenin düşey kenar uzunluğunu $(b)$ tutmaktadır. MBR'ler üzerinden önce, nesneler arasında *sahip* ilişkisi incelenerek (satır 6-9), sonra aralarında bu ilişki bulunmayan nesneler arasında *teğet* ve *yakın* ilişkileri (satır 11-16) ile *yönsel* ilişkiler (satır 17 ve 18) belirlenmektedir.

**Algoritma 7.1:** $uzamsal\_cikarsama(MBR)$

**Girdiler:** Nesneleri temsil eden en küçük kapsayan dikdörtgenler ($MBR$)
**Çıktılar:** Nesneler arasında belirlenen topolojik ilişkiler listesi ($\mathcal{I}$) ve
her bir nesne çifti arasındaki yönsel ilişkileri tutan matrisler ($\mathcal{I}_Y$)

1: $n \leftarrow boyut(MBR)$
2: **her** $1 \leq i < n$ **için**
3: **her** $i < j \leq n$ **için**
4: $N_i \leftarrow MBR_i(1), x_i \leftarrow MBR_i(2), y_i \leftarrow MBR_i(3), a_i \leftarrow MBR_i(4), b_i \leftarrow MBR_i(5)$
5: $N_j \leftarrow MBR_j(1), x_j \leftarrow MBR_j(2), y_j \leftarrow MBR_j(3), a_j \leftarrow MBR_j(4), b_j \leftarrow MBR_j(5)$
6: **eğer** $x_i \leq x_j$ **ve** $x_i + a_i \geq x_j + a_j$ **ve** $y_i \leq y_j$ **ve** $y_i + b_i \geq y_j + b_j$ **ise**
7: $ekle$ "$sahip(N_1, N_2)$" $\rightarrow \mathcal{I}$
8: **değilse ve** $(x_j \leq x_i$ **ve** $x_j + a_j \geq x_i + a_i$ **ve** $y_j \leq y_i$ **ve** $y_j + b_j \geq y_i + b_i)$ **ise**
9: $ekle$ "$sahip(N_2, N_1)$" $\rightarrow \mathcal{I}$
10: **değilse**
11: $uzaklik \leftarrow uzaklik\_hesapla\big(MBR(i), MBR(j)\big)$
12: **eğer** $uzaklik < esik\_deger$ **ise**
13: **eğer** $uzaklik = 0$ **ise**
14: $ekle$ "$teğet(N_1, N_2)$" $\rightarrow \mathcal{I}$
15: **degilse**
16: $ekle$ "$yakın(N_1, N_2)$" $\rightarrow \mathcal{I}$
17: $\mathcal{I}_Y\big(N_i, N_j\big) \leftarrow yonsel\_iliskileri\_belirle\big(MBR(i), MBR(j)\big)$
18: $\mathcal{I}_Y\big(N_j, N_i\big) \leftarrow yonsel\_iliskileri\_belirle\big(MBR(j), MBR(i)\big)$
19: **döndür** $\mathcal{I}, \mathcal{I}_Y$

İki nesnenin en küçük kapsayan dikdörtgenleri arasındaki uzaklık, Algoritma 7.2'de gösterilen adımlarla hesaplanmaktadır. Öklid uzaklık değerinin hesaplanması (satır 15) için, öncelikle, en küçük kapsayan dikdörtgenlerin yatay ve düşey izdüşümleri olan doğrular arasında yatay (satır 3-8) ve düşey uzaklıklar belirlenmektedir (satır 9-14). Kesişen doğrular arasındaki uzaklık sıfırdır (satır 8 ve satır 14).

**Algoritma 7.2:** $uzaklik\_hesapla(MBR_1, MBR_2)$

**Girdiler:** İki nesneyi temsil eden en küçük kapsayan dikdörtgenler ($MBR_1$ ve $MBR_2$)
**Çıktılar:** En küçük kapsayan dikdörtgenler arasında hesaplanan uzaklık ($U$)

$x_1 \leftarrow MBR_1(2)$, $y_1 \leftarrow MBR_1(3)$, $a_1 \leftarrow MBR_1(4)$, $b_1 \leftarrow MBR_1(5)$
$x_2 \leftarrow MBR_2(2)$, $y_2 \leftarrow MBR_2(3)$, $a_2 \leftarrow MBR_2(4)$, $b_2 \leftarrow MBR_2(5)$
**eğer** $x_1 > x_2 + a_2$ **ise**
    $yatay\_uzaklık \leftarrow (x_1 - (x_2 + a_2))$
**değilse ve** $x_2 > x_1 + a_1$ **ise**
    $yatay\_uzaklık \leftarrow (x_2 - (x_1 + a_1))$
**değilse**
    $yatay\_uzaklık \leftarrow 0$
**eğer** $y_1 > y_2 + b_2$ **ise**
    $dusey\_uzaklık \leftarrow (y_1 - (y_2 + b_2))$
**değilse ve** $y_2 > y_1 + b_1$ **ise**
    $dusey\_uzaklık \leftarrow (y_2 - (y_1 + b_1))$
**değilse**
    $dusey\_uzaklık \leftarrow 0$
$U \leftarrow \sqrt{yatay\_uzaklık^2 + dusey\_uzaklık^2}$
**döndür** $U$

İki nesne arasındaki yönsel ilişkiler, nesnelerin en küçük kapsayan dikdörtgenleri üzerinden Algoritma 7.3'teki adımlarla belirlenmektedir. Yönsel ilişkileri ifade etmek için 3x3 boyutunda bir matris kullanılmaktadır. Bu matris, en küçük kapsayan dikdörtgenlerin düşey ve yatay izdüşümleri arasındaki yönsel ilişkileri belirten vektörlerin dış çarpımı (*outer product*) ile elde edilmektedir.

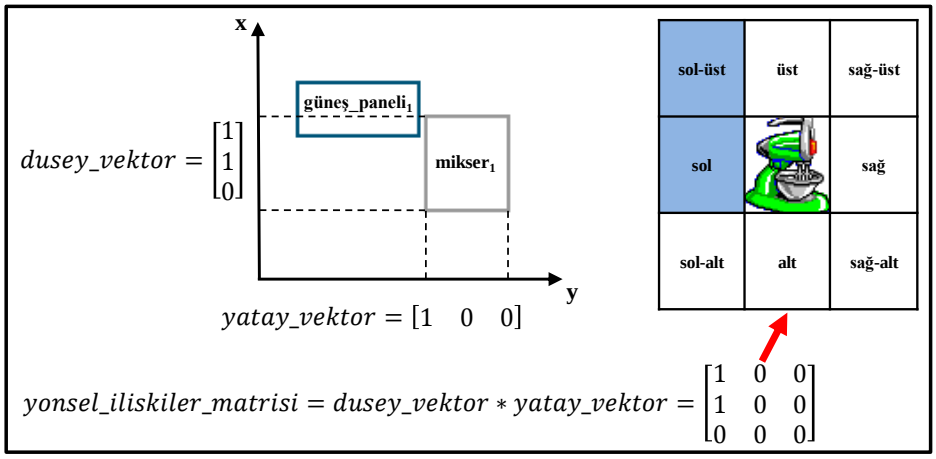
**Algoritma 7.3:** $yonsel\_iliskileri\_belirle(MBR_1, MBR_2)$

**Girdiler:** İki nesneyi temsil eden en küçük kapsayan dikdörtgenler ($MBR_1$ ve $MBR_2$)
**Çıktılar:** Yönsel ilişkiler matrisi ($YM$)

$x_1 \leftarrow MBR_1(2)$, $y_1 \leftarrow MBR_1(3)$, $a_1 \leftarrow MBR_1(4)$, $b_1 \leftarrow MBR_1(5)$
$x_2 \leftarrow MBR_2(2)$, $y_2 \leftarrow MBR_2(3)$, $a_2 \leftarrow MBR_2(4)$, $b_2 \leftarrow MBR_2(5)$
$yatay\_vektor \leftarrow yonsel\_vektor\_olustur(x_1, a_1, x_2, a_2)$
$dusey\_vektor \leftarrow yonsel\_vektor\_olustur(y_1, b_1, y_2, b_2)$
$YM \leftarrow dusey\_vektor^T * yatay\_vektor$
**döndür** $YM$

Şekil 7.6'da, örnek eğitim senaryosundaki $güneş\_paneli_1$ nesnesinin $mikser_1$ nesnesine göre bağıl konumunun yatay ve düşey yönsel vektörler yardımıyla nasıl ifade edildiği gösterilmektedir. Yatay doğrultuda $güneş\_paneli_1$ $mikser_1$'in sol tarafındadır. Düşey doğrultuda ise, $güneş\_paneli_1$'in bir kısmı $mikser_1$ ile aynı

hizada bir kısmı ise $mikser_1$'den yukarıdadır. Bu durumu modellemek için oluşturulan $YM$ yönsel ilişkiler matrisinde sol-üst bölgeye karşı düşen $YM_{1,1}$ ve sol bölgeye karşı düşen $YM_{2,1}$ gözleri 1 diğer gözler ise 0 değerini almaktadır.




**Şekil 7.6:** İkili yönsel ilişkilerin vektörlerle ifade edilmesi.

Düşey ve yatay yönlerdeki bağıl konum bilgilerini elde ederken kullanılan tek boyutlu yönsel ilişkiler, Algoritma 7.4'te gösterilen şekilde elde edilmektedir. Paralel iki doğru parçası arasında 6 durum söz konusu olabilir: 1. doğru parçası 2. doğru parçasının solunda olabilir (satır 1 ve 2), içinde olabilir (satır 3 ve 4), sağında olabilir (satır 5 ve 6), solundan çakışık olabilir (satır 7 ve 8), sağından çakışık olabilir (satır 9 ve 10) veya 1. doğru parçası 2. doğru parçasını kapsayabilir (satır 11 ve 12).

**Algoritma 7.4:** $yonsel\_vektor\_olustur(nokta_1, uzunluk_1, nokta_2, uzunluk_2)$

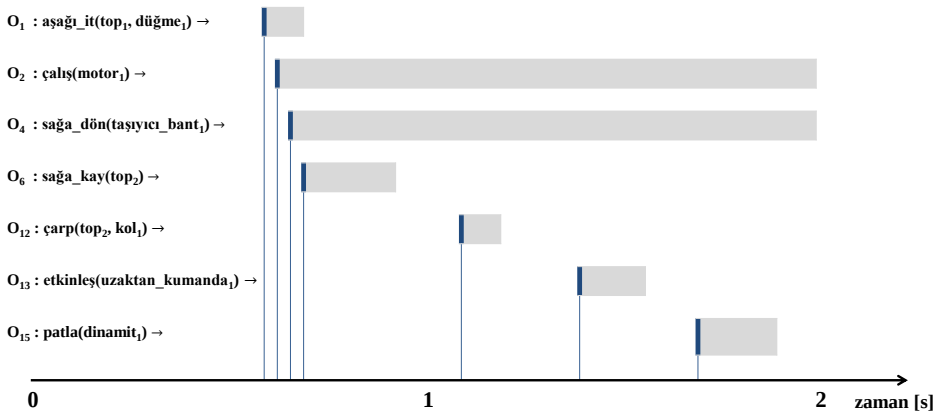
**Girdiler:** Birbirine paralel iki doğru parçasının başlangıç noktaları ($nokta_1$ ve $nokta_2$) ve uzunlukları ($uzunluk_1$ ve $uzunluk_2$)
**Çıktılar:** 1. doğru parçasının, 2. doğru parçasına göre yönsel ilişkisini temsil eden vektör ($YV$)

1: **eğer** $nokta_2 \geq nokta_1 + uzunluk_1$ **ise**
2: $YV \leftarrow [1 \quad 0 \quad 0]$
3: **değilse ve** ($nokta_1 \geq nokta_2$ **ve** $nokta_2 + uzunluk_2 \geq nokta_1 + uzunluk_1$) **ise**
4: $YV \leftarrow [0 \quad 1 \quad 0]$
5: **değilse ve** ($nokta_1 \geq nokta_2 + uzunluk_2$ ) **ise**
6: $YV \leftarrow [0 \quad 0 \quad 1]$
7: **değilse ve** ($nokta_1 < nokta_2 < (nokta_1 + uzunluk_1) \leq (nokta_2 + uzunluk_2)$) **ise**
8: $YV \leftarrow [1 \quad 1 \quad 0]$
9: **değilse ve** ($nokta_2 \leq nokta_1 < (nokta_2 + uzunluk_2) < (nokta_1 + uzunluk_1)$) **ise**
10: $YV \leftarrow [0 \quad 1 \quad 1]$
11: **değilse ve** ($nokta_1 < nokta_2$ **ve** $(nokta_1 + uzunluk_1) > (nokta_2 + uzunluk_2)$ ) **ise**
12: $YV \leftarrow [1 \quad 1 \quad 1]$
13: **döndür** $YV$

## 7.4 Zamansal Çıkarsama

Nesnelere ilişkin uzamsal bilgiler sisteme girdi olarak sağlanmadığında, ortamdaki nesneler üzerinde gerçekleşen olayların zaman bilgilerini kullanan zamansal bir yaklaşımla ilişkilerin öğrenilmesi analiz edilmiştir. Bu analizde, her bir olayın başlangıç zamanı olay listesine eklenerek sisteme girdi olarak verilmiştir. Nokta-zaman aralığı cebrindeki (Vilain M. B., 1982) *begins* olgusuna benzer şekilde, eğer bir $O_j$ olayı bir $O_i$ olayının çok kısa bir süre ardından gerçekleşiyorsa, bu olayların her bir argümanı arasında zamanla aynı yönde ilişki ($Arg_{1:p_i}(O_i) \rightarrow Arg_{1:p_j}(O_j)$) kurulabilir.

Şekil 7.1'deki örnek eğitim senaryosunda $O_1$, $O_2$, $O_4$, $O_6$, $O_{12}$, $O_{13}$ ve $O_{15}$ olaylarının oluşturduğu zincirleme etkileşim Şekil 7.7'deki zaman diyagramında verilmiştir. Birbirleri ile etkileşim içinde olan nesneler üzerinde gerçekleşen olaylar arasında çok kısa bir zaman farkının olduğunun görüldüğü bu etkileşim zincirinde, $top_1$'in $düğme_1$'i aşağı itmesi ile $düğme_1$ tarafından kontrol edilen prize takılı olan $motor_1$ çalışmaya başlamaktadır. $motor_1$ çalışınca ona kayışla bağlı olan $taşıyıcı\_bant_1$'in sağa dönmesini sağlamakta ve $taşıyıcı\_bant_1$ sağa döndüğünde üzerindeki $top_2$ de sağa kaymaktadır. Sağa kayan $top_2$, $dinamit_1$'i kontrol eden uzaktan kumanda kolunun üzerine düşerek uzaktan kumandayı etkinleştirmekte ve $dinamit_1$ patlamaktadır. Bu olaylar üzerinden, olay argümanlarındaki nesneler için belirlenen zamansal ilişkiler Çizelge 7.3'te gösterilmektedir. Bu örnek, nesneler arası bağlantılara ilişkin bilgiye sahip olunmadığında olay zamanları üzerinden çıkarsama yapmanın mümkün olduğunu göstermektedir.




**Şekil 7.7:** Zincirleme etkileşimdeki olaylar arasındaki zamansal ilişkiler.

**Çizelge 7.3:** Şekil 7.7'deki olaylar üzerinden çıkarsanan zamansal ilişkiler.

| İlişki ifadesi | Etki yönü |
|---|---|
| *tetikler(top$_1$,motor$_1$)* | |
| *tetikler(düğme$_1$,motor$_1$)* | |
| *tetikler(motor$_1$,taşıyıcı_bant$_1$)* | |
| *tetikler(taşıyıcı_bant$_1$,top$_2$)* | $n_1 \rightarrow n_2$ |
| *tetikler(top$_2$,kol$_1$)* | |
| *tetikler(top$_2$,uzaktan_kumanda$_1$)* | |
| *tetikler(kol$_1$,uzaktan_kumanda$_1$)* | |
| *tetikler(uzaktan_kumanda$_1$,dinamit$_1$)* | |

### 7.4.1 Zamansal çıkarsama algoritması

Zamansal çıkarsama, Algoritma 7.5'te gösterilen adımlarla yapılmaktadır. Girdi olarak alınan olayların sıralı dizisindeki ($\mathcal{O}$) her bir olay için takip eden sıradaki olay incelenmekte ve eğer olayların başlangıç zamanları arasında belirlenen eşik değerin altında bir fark söz konusu ise olayların her bir argümanı arasında zamanla aynı yönde ilişkiler tanımlanmaktadır (satır 6-11). Eş zamanlı olaylarda, bu olayların önceki olayla aralarındaki zaman farkı eşik değerin altında ise önceki olayın tümünü tetiklediği varsayılmaktadır (satır 12-18).

**Algoritma 7.5:** $zamansal\_cikarsama(\mathcal{O}, \mathcal{Z})$

**Girdiler:** Olayların sıralı dizisi ($\mathcal{O}$), $\mathcal{O}$'daki her bir olayın başlangıç zamanı ($\mathcal{Z}$)
**Çıktılar:** Nesneler arasında belirlenen zamansal ilişkiler ($\mathcal{I}$)

1: $n \leftarrow boyut(\mathcal{O})$
2: **her** $1 \leq i < n$ **için**
3: $\quad j \leftarrow i + 1$
4: $\quad \mathcal{Z}(O_j) = \mathcal{Z}(O_i)$ **olduğu sürece**
5: $\quad\quad j \leftarrow j + 1$
6: $\quad$ **eğer** $\mathcal{Z}(O_j) - \mathcal{Z}(O_i) < esik\_deger$ **ise**
7: $\quad\quad$ **her** $1 \leq k \leq p_i$ **için**
8: $\quad\quad\quad N_1 \leftarrow O_{i,k}$
9: $\quad\quad\quad$ **her** $1 \leq l \leq p_j$ **için**
10: $\quad\quad\quad\quad N_2 \leftarrow O_{j,l}$
11: $\quad\quad\quad\quad ekle\ "tetikler(N_1, N_2)" \rightarrow \mathcal{I}$
12: $\quad\quad \mathcal{Z}(O_j) = \mathcal{Z}(O_{j+1})$ **olduğu sürece**
13: $\quad\quad\quad j \leftarrow j + 1$
14: $\quad\quad\quad$ **her** $1 \leq k \leq p_i$ **için**
15: $\quad\quad\quad\quad N_1 \leftarrow O_{i,k}$
16: $\quad\quad\quad\quad$ **her** $1 \leq l \leq p_j$ **için**
17: $\quad\quad\quad\quad\quad N_2 \leftarrow O_{j,l}$
18: $\quad\quad\quad\quad\quad ekle\ "tetikler(N_1, N_2)" \rightarrow \mathcal{I}$
19: **döndür** $\mathcal{I}$

### 7.5 Uzam-zamansal Çıkarsama

*The Incredible Machine* oyununda, insan seviyesinde öğrenmede uzamsal ve zamansal bilgi birlikte kullanılarak çözüm üretilmektedir. Benzer şekilde önerilen sistemde de uzamsal ve zamansal yaklaşımların iyi yönlerini birleştiren uzam-zamansal bir yaklaşımla daha iyi sonuçlar elde edilebileceği düşünülerek zamansal çıkarsamaya dayalı yöntem içinde nesnelerin uzam bilgilerini de kullanan uzam-zamansal bir yaklaşım geliştirilmiştir. Genel olarak zamansal çıkarsama daha iyi sonuçlar vermesine rağmen, bazı durumlar için uzamsal çıkarsama daha iyi sonuçlar doğurmuştur. Örneğin, *begins* örüntüsünü sağlayan iki olayın hangi argümanlarının birbiri ile ilişkili olduğunu belirlemek üzere uzamsal çıkarsamadan faydalanılabilen durumlar bulunmaktadır. Birbirine yakın nesneler üzerindeki çok argümanlı olaylarda uzamsal yöntemin sonuçları kullanılarak başarım artırılmıştır. Ayrıca, Şekil 7.1'deki $ekmek\_kızartıcı_1$ nesnesi gibi çalışabilmesi için birden fazla önkoşul gereken (prize takılı olmalı, priz açık olmalı ve tost makinesinin düğmesine basılmış olmalı) nesnelerin modellenmesinde de uzamsal çıkarsamayla birden fazla önkoşul üretilmişse bu sonuçlar kullanılmıştır. Örnek olarak, Şekil 7.7'deki zincirleme etkileşim halindeki olaylar için olay argümanlarındaki nesneler için belirlenen uzam-zamansal ilişkiler Çizelge 7.4'te gösterilmektedir.

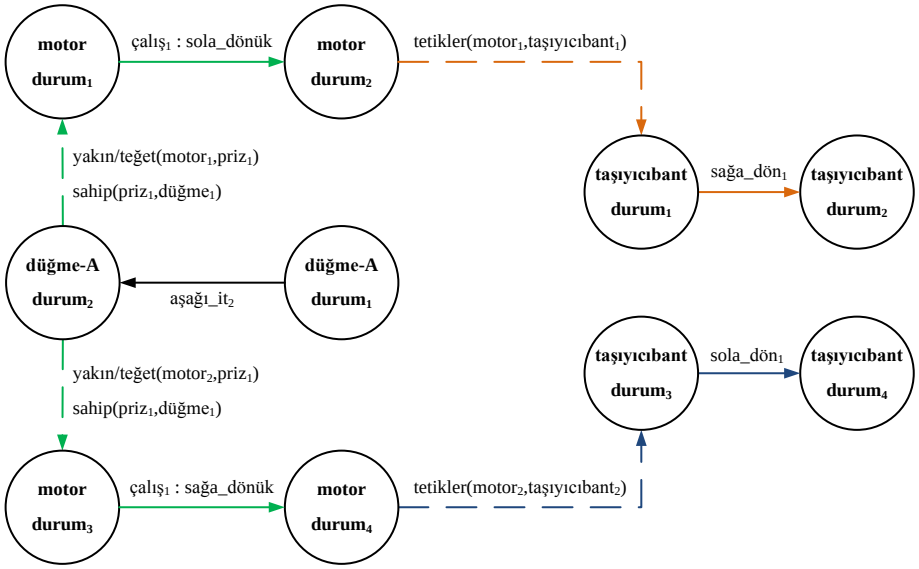
**Çizelge 7.4:** Şekil 7.7'deki olaylara ilişkin uzam-zamansal ilişkiler.

| **İlişki ifadesi** | **Şekil 7.7'den örnek** | **Etki yönü** |
|---|---|---|
| *sahip($n_1$,$n_2$)* | *sahip($priz_1$,$düğme_1$)*<br>*sahip($uzaktan\_kumanda_1$,$kol_1$)* | $n_1 \leftrightarrow n_2$ |
| *yakın($n_1$,$n_2$)* | *yakın($motor_1$,$priz_1$)* | |
| *tetikler($n_1$,$n_2$)* | *tetikler($motor_1$,$taşıyıcı\_bant_1$)*<br>*tetikler($taşıyıcı\_bant_1$,$top_2$)*<br>*tetikler($top_2$,$kol_1$)*<br>*tetikler($uzaktan\_kumanda_1$,$dinamit_1$)* | $n_1 \rightarrow n_2$ |

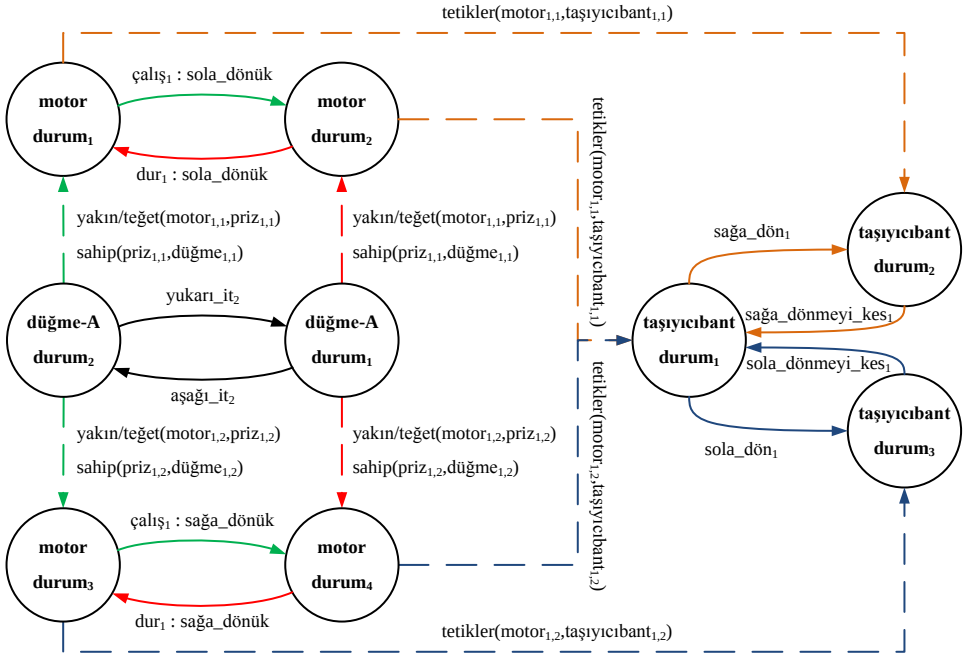
Uzam-zamansal çıkarsamada aralarında uzamsal ilişki bulunmaksızın sadece zamansal ilişki kurulan nesnelerin etkileşime geçebilmeleri için sağlanması gereken ortamdaki birbirlerine göre bağıl konumlarının öğrenilmesi için uzamsal yaklaşımdaki gibi ana yönler modelinden (Frank, 1991) faydalanılmıştır.

Örnek eğitim senaryosundaki $düğme_1$, $motor_1$, $motor_2$, $taşıyıcı\_bant_1$ ve $taşıyıcı\_bant_2$ nesneleri için sadece bu senaryoyu kullanarak uzam-zamansal çıkarsamayla bulunan etkileşimler Şekil 7.8'de verilmiştir. Aynı nesnelerin farklı

etkileşimlerini barındıran Şekil 6.1 ve 6.4'teki eğitim senaryoları kullanılarak uzam-zamansal yaklaşımla öğrenilen sonlu durum makineleri Şekil 7.9'daki son halini almaktadır. Öğrenme sırasında, bu nesnelerden *düğme* ve *priz* arasında bilgi tabanlı yaklaşımla aynı şekilde *sahip* ilişkisi kurulmuştur. *motor* ve *priz* arasında ise *teğet* ya da *yakın* ilişkisi üzerinden bir etkileşim söz konusu olabilmektedir. *motorun taşıyıcı_bandı* sürdüğü örneklerde sadece zamansal bir ilişki (*tetikler*) söz konusudur. Bu ilişkilerde ana yönler modeliyle elde edilen bağıl konumlar önemli değildir.




**Şekil 7.8:** Şekil 7.1'deki senaryoda uzam-zamansal çıkarsamanın sonuçları.




**Şekil 7.9:** Şekil 6.1 ve 6.4'teki senaryolarda uzam-zamansal çıkarsama sonuçları.

### 7.5.1 Uzam-zamansal çıkarsama algoritması

Uzam-zamansal çıkarsama ile nesneler arasındaki ilişkiler belirlenirken Algoritma 7.6'da gösterilen adımlar uygulanmaktadır. Eğitim senaryolarında birbirinden bağımsız eş zamanlı olay zincirlerinin olabileceği (örneğin, Şekil 7.1'deki senaryodaki $motor_1$ ve $motor_2$ tarafından başlatılan iki etkileşim zinciri ve $top_1$'in $el\_feneri_1$'i kontrol eden $düğme_3$'e basması ile başlayan üçüncü etkileşim zinciri) düşünülerek önce bu zincirler içinde birbirini tetikleyen olaylar uzamsal ilişkilerden yararlanılarak belirlenmektedir (Algoritma 7.7). Sonra, birbirleri ile zamansal ilişkiye sahip olduğu belirlenen olay çiftleri arasında uzamsal çıkarsamanın zamansal çıkarsamaya göre avantajlı olduğu durumlarda (argüman seçimi ve birden fazla önkoşulu olan olayların belirlenmesi) uzamsal çıkarsama ile belirlenen ilişkilerin tercih edildiği Algoritma 7.8'deki adımlar yürütülmektedir.

---

**Algoritma 7.6:** $uzam\_zamansal\_cikarsama(\mathcal{O}, \mathcal{Z}, MBR, uzamsal)$

---

**Girdiler:** Olayların sıralı dizisi ($\mathcal{O}$), $\mathcal{O}$'daki her bir olayın başlangıç zamanı ($\mathcal{Z}$),
nesneleri temsil eden en küçük kapsayan dikdörtgenler ($MBR$),
nesneler arasındaki uzamsal ilişkiler listesi ($uzamsal$)

**Çıktılar:** Nesneler arasında belirlenen uzam-zamansal ilişkiler listesi ($\mathcal{I}$) ve
zamansal ilişkili nesne çiftleri arasındaki yönsel ilişkileri tutan matrisler ($\mathcal{I}_Y$)

1: $zamansal \leftarrow zamansal\_iliskileri\_belirle(\mathcal{O}, \mathcal{Z}, MBR)$

2: $\mathcal{I}, \mathcal{I}_Y \leftarrow uzam\_zamansal\_iliskileri\_belirle(zamansal, uzamsal, MBR)$

3: **döndür** $\mathcal{I}, \mathcal{I}_Y$

---

Algoritma 7.7'deki adımlar uygulanarak olay listesindeki her bir olay için o olay öncesinde gerçekleşen olaylardan hangisinin o olayı başlattığı belirlenmeye çalışılır. Eş zamanlı olay zincirleri arasında bir olayın nasıl tetiklendiğini belirlemek üzere önce ortak argümanlı olaylara bakılır (satır 5-14). Eğer bir olay öncesinde o olayın argümanındaki nesnelerden herhangi birini etkileyen başka bir olay yoksa zamansal ilişkili olaylardan en yakın konumda (Algoritma 7.2'deki uzaklık hesaplama) gerçekleşen seçilir (satır 15-27). Zamansal ilişkileri belirlemek için olay zincirleri üzerinden yapılan çıkarsamayla elde edilen zincir sonundaki olaylara ilişkin bilgi bir listede tutulur (satır 2,12,26 ve 29). Yeni olaylar zincire eklendikçe, zincir sonlarını tutan listeden silinecek önceki olaylar belirlenerek (satır 14 ve 27) sıradaki incelenen olay bir öncekiyle eş zamanlı değilse silinecek olaylar zincir sonu listesinden çıkarılır (satır 30-33).

**Algoritma 7.7:** $zamansal\_iliskileri\_belirle(\mathcal{O}, \mathcal{Z}, MBR)$

**Girdiler:** Olayların sıralı dizisi ($\mathcal{O}$), $\mathcal{O}$'daki her bir olayın başlangıç zamanı ($\mathcal{Z}$), nesneleri temsil eden en küçük kapsayan dikdörtgenler ($MBR$)
**Çıktılar:** Olaylar arasında belirlenen zamansal ilişkiler ($zamansal$)

```
1:  zincir_sonu ← ∅, silinecek_zincir_sonu ← ∅
2:  ekle O_1 → zincir_sonu
3:  n ← boyut(O)
4:  her 1 < i ≤ n için
5:      parametrik_iliski ← false
6:      !parametrik_iliski olduğu sürece her 1 ≤ k ≤ p_i için
7:          her i > j ≥ 1 için
8:              her 1 ≤ l ≤ p_j için
9:                  eğer O_{i,k} = O_{j,l} ise
10:                     ekle (O_j, O_i) → zamansal
11:                     parametrik_iliski ← true
12:                     ekle O_i → zincir_sonu
13:                     eğer O_j ∈ zincir_sonu ise
14:                         ekle O_j → silinecek_zincir_sonu
15:     eğer !parametrik_iliski ise
16:         zamansal_iliski ← false, min_uzaklık ← ∞
17:         her O_j ∈ zincir_sonu için
18:             eğer Z(O_j) ≠ Z(O_i) ve Z(O_j) − Z(O_i) < esik_deger ise
19:                 her 1 ≤ k ≤ p_i için
20:                     her 1 ≤ l ≤ p_j için
21:                         uzaklik ← uzaklik_hesapla(MBR(O_{i,k}), MBR(O_{j,l}))
22:                         eğer uzaklik < min_uzaklik ise
23:                             eklenecek ← O_j, min_uzaklik ← uzaklik
24:         ekle (eklenecek, O_i) → zamansal
25:         zamansal_iliski ← true
26:         ekle O_i → zincir_sonu
27:         ekle eklenecek → silinecek_zincir_sonu
28:     eğer !parametrik_iliski ve !zamansal_iliski ise
29:         ekle O_i → zincir_sonu
30:     eğer Z(O_i) ≠ Z(O_{i+1}) ise
31:         her O_j ∈ silinecek_zincir_sonu için
32:             sil zincir_sonu(O_j)
33:         silinecek_zincir_sonu ← ∅
34: döndür zamansal
```

Zamansal ilişkili olduğu belirlenen tüm olay çiftleri arasında Algoritma 7.8'deki adımlar uygulanarak nesneler arasındaki uzam-zamansal ilişkiler elde edilmektedir. Öncelikle sistemin birden fazla ön koşula sahip olan olayları modelleyebilmesi için birden fazla uzamsal ilişkiye sahip olan nesnelerde, bu ilişkiler zamansal çıkarsamanın sonucuna tercih edilmektedir (satır 6-8 ve 16-19). Bunun dışında, tek

argümanlı olaylar tarafından etkilenen nesneler arasında zamansal ilişki kullanılırken (satır 10 ve 11), çok argümanlı olaylarda ilişkili argümanların belirlenmesi için uzamsal çıkarsamayla elde edilen ilişkilerden yararlanılmaktadır (satır 23-26). Aralarında sadece zamansal ilişki bulunan nesnelerde, yönsel ilişki bilgisi de (Algoritma 7.3 kullanılarak belirlenen) tutulmaktadır (satır 11 ve satır 32).

**Algoritma 7.8:** $uzam\_zamansal\_iliskileri\_belirle(zamansal, uzamsal, MBR)$

**Girdiler:** Olaylar arasındaki zamansal ilişkiler ($zamansal$),
nesneler arasındaki uzamsal ilişkiler ($uzamsal$),
nesneleri temsil eden en küçük kapsayan dikdörtgenler ($MBR$)
**Çıktılar:** Nesneler arasında belirlenen uzam-zamansal ilişkiler listesi ($\mathcal{I}$) ve
zamansal ilişkili nesne çiftleri arasındaki yönsel ilişkileri tutan matrisler ($\mathcal{I}_Y$)

1: $n \leftarrow boyut(zamansal)$
2: **her** $1 \leq i \leq n$ **için**
3: $O_1 \leftarrow zamansal(i,1),\ O_2 \leftarrow zamansal(i,2)$
4: **eğer** $p_1 = 1$ **ve** $p_2 = 1$ **ise**
5: $N_1 \leftarrow O_{1,1},\ N_2 \leftarrow O_{2,1}$
6: $U_2 \leftarrow coklu\_uzamsal\_iliskileri\_belirle(uzamsal, N_2)$
7: **eğer** $boyut(U_2) \geq 2$ **ise**
8: $ekle\ U_2 \rightarrow \mathcal{I}$
9: **değilse**
10: $ekle$ "$tetikler(N_1, N_2)$" $\rightarrow \mathcal{I}$
11: $\mathcal{I}_Y(N_1, N_2) \leftarrow yonsel\_iliskileri\_belirle\big(MBR(N_1), MBR(N_2)\big)$
12: **değilse**
13: $uzamsal\_iliski\_bulundu \leftarrow false$
14: **her** $1 \leq l \leq p_2$ **için**
15: $N_2 \leftarrow O_{2,l}$
16: $U_2 \leftarrow coklu\_uzamsal\_iliskileri\_belirle(uzamsal, N_2)$
17: **eğer** $boyut(U_2) \geq 2$ **ise**
18: $ekle\ U_2 \rightarrow \mathcal{I}$
19: $uzamsal\_iliski\_bulundu \leftarrow true$
20: **değilse**
21: **her** $1 \leq k \leq p_1$ **için**
22: $N_1 \leftarrow O_{1,k}$
23: $U_1 \leftarrow uzamsal\_iliskileri\_belirle(uzamsal, N_1, N_2)$
24: **eğer** $U_1 \neq \emptyset$ **ise**
25: $ekle\ U_1 \rightarrow \mathcal{I}$
26: $uzamsal\_iliski\_bulundu \leftarrow true$
27: **eğer** $!uzamsal\_iliski\_bulundu$ **ise**
28: **her** $1 \leq k \leq p_1$ **için**
29: **her** $1 \leq l \leq p_2$ **için**
30: $N_1 \leftarrow O_{1,k},\ N_2 \leftarrow O_{2,l}$
31: $ekle$ "$tetikler(N_1, N_2)$" $\rightarrow \mathcal{I}$
32: $\mathcal{I}_Y(N_1, N_2) \leftarrow yonsel\_iliskileri\_belirle\big(MBR(N_1), MBR(N_2)\big)$
33: **döndür** $\mathcal{I}, \mathcal{I}_Y$

Birden fazla uzamsal ilişkiye sahip olan nesneler üzerinde gerçekleşen çoklu ön koşullu olayların modellenebilmesi için, Algoritma 7.9'daki adımlar uygulanarak girdi olarak verilen bir nesnenin tüm uzamsal ilişkileri belirlenmektedir. Uzamsal ilişkiler listesindeki her bir ilişki için uzamsal ilişkileri belirlenmek istenen nesneyle ilgili olanlar bir listeye eklenerek (satır 5 ve 6), tüm uzamsal ilişkilerde ilgili nesne tarandıktan sonra bu liste sonuç olarak döndürülmektedir. Eğer nesneyi argüman olarak alan herhangi bir uzamsal ilişki bulunamazsa, sonuç olarak boş küme (∅) döndürülmektedir (satır 1).

**Algoritma 7.9:** $coklu\_uzamsal\_iliskileri\_belirle(\mathcal{I}, N)$

**Girdiler:** Nesneler arasındaki uzamsal ilişkiler ($\mathcal{I}$),
sahip olduğu uzamsal ilişkilerin belirleneceği nesne ($N$)
**Çıktılar:** $N$ nesnesinin sahip olduğu uzamsal ilişkiler listesi ($U$)

1: $U \leftarrow \emptyset$
2: **her** $I_i \in \mathcal{I}$ **için**
3: $N_1 \leftarrow I_{i,1}$
4: $N_2 \leftarrow I_{i,2}$
5: **eğer** $N_1 = N$ **veya** $N_2 = N$ **ise**
6: $ekle\ \mathcal{I}(I_i) \rightarrow U$
7: **döndür** $U$

İki nesne arasında uzamsal bir ilişki olup olmadığı bilgisine uzamsal ilişkiler listesini tarayarak ulaşmak üzere Algoritma 7.10'da sözde kodu verilen fonksiyondan yararlanılmıştır. Bu fonksiyonda, uzamsal ilişkiler listesindeki tüm kayıtlar taranarak argümanlarından biri ilk nesne diğeri de ikinci nesne olan bir ilişki bulunursa, sonuç olarak bu ilişki döndürülmektedir (satır 3 ve 4). Aralarında hiçbir uzamsal ilişki bulunmayan nesneler için boş küme (∅) döndürülmektedir (satır 1).

**Algoritma 7.10:** $uzamsal\_iliskileri\_belirle(\mathcal{I}, N_1, N_2)$

**Girdiler:** Nesneler arasındaki uzamsal ilişkiler ($\mathcal{I}$),
aralarındaki uzamsal ilişkinin belirleneceği nesne çifti ($N_1$ ve $N_2$)
**Çıktılar:** $N_1$ ve $N_2$ nesneleri arasındaki uzamsal ilişki ($U$)

1: $U \leftarrow \emptyset$
2: **her** $I_i \in \mathcal{I}$ **için**
3: **eğer** ($N_1 = I_{i,1}$ **ve** $N_2 = I_{i,2}$) **veya** ($N_2 = I_{i,1}$ **ve** $N_1 = I_{i,2}$) **ise**
4: $U \leftarrow \mathcal{I}(I_i)$
5: **döngüyü sonlandır**
6: **döndür** $U$

### 7.6 Nesne Etkileşimlerini Öğrenme Algoritması

Ortamdaki nesneler arasındaki ilişkileri (insan seviyesinde doğrudan gözlemlenebilir ilişkiler, uzamsal ilişkiler, zamansal ilişkiler ve uzam-zamansal ilişkiler) tutan bir bilgi tabanı kullanılarak, nesneler arasında olaylar yoluyla gerçekleşen koşullu etkileşimlerin öğrenilebilmesi için geliştirilen yaklaşımın genel mantığı Algoritma 7.11'de verilmiştir. Nesneler arası ilişkileri tutan bir bilgi tabanı ($\mathcal{I}$) ve ikili yönsel ilişkileri modelleyen matrisler ($\mathcal{I}_Y$) ile birlikte sisteme girdi olarak, olayların sıralı dizisi ($\mathcal{O}$), nesnelere ilişkin yönelim bilgileri ($\mathcal{Y}$) ve sistemin ilk aşamasında oluşturulan nesneler kümesi ($\mathcal{NK}$) verilmektedir. Bu girdiler kullanılarak, öncelikle her bir nesne için bağlantı alt kümeleri oluşturulmaktadır (satır 1). Bağlantı alt kümeleri üzerinde bağlantılar arasındaki geçişlilik özelliğinden yararlanılmakta (satır 2) ve olay kümesinde bulunmayan nesneler bu alt kümelerden çıkarılmaktadır (satır 3). Ortamdaki herhangi bir nesne için etkileşimler bulunurken, bu nesneye ait olan bağlantı alt kümelerindeki nesneler üzerinde en son gözlemlenen olayın kullanımına dayalı sezgisel bir yöntem kullanılmaktadır (satır 4).

**Algoritma 7.11:** $nesne\_etkilesimlerini\_modelle(\mathcal{I}, \mathcal{I}_Y, \mathcal{O}, \mathcal{Y}, \mathcal{NK})$

**Girdiler:** Nesneler arası ilişkileri tutan bilgi tabanı ($\mathcal{I}$), yönsel ilişki matrisleri ($\mathcal{I}_Y$), olayların sıralı dizisi ($\mathcal{O}$), nesne yönelim bilgisi ($\mathcal{Y}$), nesneler kümesi ($\mathcal{NK}$)
**Çıktılar:** Nesneler arası koşullu etkileşimler ($\mathcal{E}$)

1: $\mathcal{B} \leftarrow baglantilari\_oku(\mathcal{I})$
2: $\mathcal{B} \leftarrow gecisli\_baglantilari\_belirle(\mathcal{B})$
3: $\mathcal{B} \leftarrow gereksiz\_baglantilari\_ele(\mathcal{B}, \mathcal{NK})$
4: $\mathcal{E} \leftarrow etkilesimleri\_belirle(\mathcal{B}, \mathcal{I}_Y, \mathcal{O}, \mathcal{Y})$
5: **döndür** $\mathcal{E}$

Bilgi tabanındaki ilişkiler kullanılarak ortamdaki nesneler arası bağlantıları modelleyen bağlantı alt kümelerinin oluşturulmasına ilişkin adımlar Algoritma 7.12'de görülmektedir. Nesneler arasındaki ilişkileri tutan bilgi tabanında ($\mathcal{I}$) bulunan her bir nesne için, bağlantı alt kümeleri o nesneyi doğrudan etkileyebilen nesneleri içerecek şekilde (bağlantıların etki yönü göz önüne alınarak) ilklendirilmektedir (satır 4-10). Bağlantı alt kümeleri oluşturulduktan sonra, bağlantılar arası geçişlilikten faydalanılarak nesneyi dolaylı olarak etkileyebilecek diğer bağlantılar da alt kümelere Algoritma 7.13'te verilen adımlarla eklenmektedir. Bu işlem sırasında bağlantı alt kümelerindeki her bir nesne grubu için uygulanabilir tüm bağlantılar bulunmakta (satır 12-18) ve bu alt kümelere eklenmektedir (satır 19-

21). Bağlantı alt kümeleri herhangi bir iterasyon sonunda sabit kalana kadar geçişli bağlantıların belirlenmesi işlemine devam edilmektedir (satır 2).

**Algoritma 7.12:** $baglantilari\_oku(\mathcal{I})$

**Girdiler:** Nesneler arası ilişkileri tutan bilgi tabanı ($\mathcal{I}$)
**Çıktılar:** Nesnelere ilişkin bağlantılar ($\mathcal{B}$)

1: **her** $I_i \in \mathcal{I}$ **için**
2: $N_1 \leftarrow I_{i,1}$
3: $N_2 \leftarrow I_{i,2}$
4: **eğer** $I_i = takili$ **veya** $I_i = ustunde$ **ise**
5: $ekle\ (N_2, [\mathcal{I}(I_i)]) \rightarrow \mathcal{B}(N_1)$
6: **eğer** $I_i = kayis$ **veya** $I_i = donuk$ **veya** $I_i = tetikler$ **ise**
7: $ekle\ (N_1, [\mathcal{I}(I_i)]) \rightarrow \mathcal{B}(N_2)$
8: **eğer** $I_i = sahip$ **veya** $I_i = ip$ **veya** $I_i = yakin$ **veya** $I_i = teget$ **ise**
9: $ekle\ (N_2, [\mathcal{I}(I_i)]) \rightarrow \mathcal{B}(N_1)$
10: $ekle\ (N_1, [\mathcal{I}(I_i)]) \rightarrow \mathcal{B}(N_2)$
11: **döndür** $\mathcal{B}$

**Algoritma 7.13:** $gecisli\_baglantilari\_belirle(\mathcal{B})$

**Girdiler:** Nesnelere ilişkin bağlantılar ($\mathcal{B}$)
**Çıktılar:** Geçişli bağlantılar ($\mathcal{B}$)

1: $farkli \leftarrow true$
2: $farkli$ **olduğu sürece**
3: $farkli \leftarrow false$
4: **her** $B_i \in \mathcal{B}$ **için**
5: $N_1 \leftarrow B_i$
6: $a_1 \leftarrow boyut\big(\mathcal{B}(N_1)\big)$
7: **her** $1 \leq j \leq a_1$ **için**
8: $e_1 \leftarrow boyut\big(\mathcal{B}(N_1, j)\big)$
9: **her** $1 \leq k \leq e_1$ **için**
10: $N_2 \leftarrow \mathcal{B}(N_1, j, k, 1)$
11: $I_2 \leftarrow \mathcal{B}(N_1, j, k, 2)$
12: **eğer** $N_2 \in \mathcal{B}$ **ise**
13: $a_2 \leftarrow boyut\big(\mathcal{B}(N_2)\big)$
14: **her** $1 \leq l \leq a_2$ **için**
15: $e_2 \leftarrow boyut\big(\mathcal{B}(N_2, l)\big)$
16: **her** $1 \leq m \leq e_2$ **için**
17: $N_3 \leftarrow \mathcal{B}(N_2, l, m, 1)$
18: $I_3 \leftarrow \mathcal{B}(N_2, l, m, 2)$
19: **eğer** $N_3 \notin \mathcal{B}(N_1, j)$ **ve** $N_1 \neq N_3$ **ise**
20: $ekle\ (N_3, [I_2, I_3]) \rightarrow \mathcal{B}(N_1, j)$
21: $farkli \leftarrow true$
22: **döndür** $\mathcal{B}$

Nesneler arasındaki etkileşimleri belirlerken nesneler üzerinde gerçekleşen olaylardan faydalanıldığı için, gözlemlenen olaylarda parametre olarak yer almayan nesnelere ilişkin bağlantıların kullanılması gereksizdir. Bu nedenle, Algoritma 7.14'te belirtilen adımlar uygulanarak, olaylar tarafından etkilenmeyen nesnelere ait ilişkiler bağlantı alt kümelerinden silinmektedir. Bu işlem sırasında, silinecek nesnelere ilişkin oluşturulan bağlantı alt kümeleri silindikten sonra (satır 3-4), bu nesneler diğer nesnelerin bağlantı alt kümelerinden de çıkarılmaktadır (satır 11-12).

**Algoritma 7.14:** $gereksiz\_baglantilari\_ele(\mathcal{B}, \mathcal{NK})$

**Girdiler:** Nesnelere ilişkin bağlantılar ($\mathcal{B}$), nesneler kümesi ($\mathcal{NK}$)
**Çıktılar:** Nesnelere ilişkin bağlantılar ($\mathcal{B}$)

```
1:  her B_i ∈ B için
2:      N_1 ← B_i
3:      eğer N_1 ∉ NK ise
4:          sil B(N_1)
5:      değilse
6:          a ← boyut(B(N_1))
7:          her 1 ≤ j ≤ a için
8:              e ← boyut(B(N_1, j))
9:              her 1 ≤ k ≤ e için
10:                 N_2 ← B(N_1, j, k, 1)
11:                 eğer N_2 ∉ NK ise
12:                     sil B(N_1, j, k)
13: döndür B
```

Nesneler arasındaki doğrudan ve dolaylı ilişkileri modelleyen bağlantı alt kümeleri oluşturulduktan sonra, bu bağlantılar ($\mathcal{B}$) ve nesneler arasındaki bağıl yönsel ilişkileri tutan matrisler ($\mathcal{J}_Y$) Algoritma 7.15'te verilen sezgisel yaklaşımla olaylar üzerinden etkileşimlerin öğrenilmesi için kullanılmaktadır. Olaylar dizisindeki ($\mathcal{O}$) her bir olayın ($O_i$) her bir parametresi ($k$) için tüm koşullu etkileşimler belirlenirken, o parametrede bulunan nesnenin ($O_{i,k}$) bağlı olduğu bütün nesneler ($\mathcal{B}(O_{i,k})$) ve aralarındaki ilişkiler dikkate alınmaktadır (satır 4-13). Tüm nesneler üzerinde en son gerçekleşen olay bilgisi (olay adı ve parametre) ve nesne yönelimine ilişkin bilgi tutulmaktadır (satır 24). Herhangi bir nesnenin etkileşimleri belirlenirken, o nesne ile ilişkili nesneler üzerinde gerçekleşen son olay kullanılmaktadır (satır 14-18). Belirlenen etkileşimlere ilişkin tutulan bilgide, etkileşime neden olan durum geçişi (olay adı ($O_i$) ve parametre ($k$)), etkileşim içindeki nesneler arasındaki bağlantı ($I$) ve varsa yönsel ilişki ($YM$) ile nesne yönelim bilgileri ($yon_1$ ve $yon_2$) bulunmaktadır.

**Algoritma 7.15:** $etkilesimleri\_belirle(\mathcal{B}, \mathcal{I}_Y, \mathcal{O}, \mathcal{Y})$

**Girdiler:** Nesnelere ilişkin bağlantılar ($\mathcal{B}$), yönsel ilişki matrisleri ($\mathcal{I}_Y$), olayların sıralı dizisi ($\mathcal{O}$), nesne yönelim bilgisi ($\mathcal{Y}$)
**Çıktılar:** Nesneler arası koşullu etkileşimler ($\mathcal{E}$)

$son\_gozlem \leftarrow \emptyset$
**her** $O_i \in \mathcal{O}$ **için**
  **her** $1 \leq k \leq p_i$ **için**
    **eğer** $O_{i,k} \in \mathcal{B}$ **ise**
      $N_1 \leftarrow O_{i,k}$
      $yon_1 \leftarrow \mathcal{Y}(N_1)$
      $a \leftarrow boyut(\mathcal{B}(N_1))$
      **her** $1 \leq l \leq a$ **için**
        $e \leftarrow boyut\big(\mathcal{B}(N_1, l)\big)$
        **her** $1 \leq m \leq e$ **için**
          $N_2 \leftarrow \mathcal{B}(N_1, l, m, 1)$
          $I \leftarrow \mathcal{B}(N_1, j, k, 2)$
          $YM \leftarrow \mathcal{I}_Y(N_2, N_1)$
          **eğer** $N_2 \in son\_gozlem$ **ise**
            **eğer** $boyut\big(E(l)\big) = 0$ **ise**
              $E(l) = (son\_gozlem(N_2), I, yon_1, YM)$
            **değilse ve** $E(l, 1) < son\_gozlem(N_2, 1)$
              $E(l) = (son\_gozlem(N_2), I, yon_1, YM)$
        **eğer** $boyut\big(E(l)\big) \neq 0$ **ise**
          $E(l) = \ E(l, 2:)$
      **eğer** $E \notin \mathcal{E}(O_i, k)$ **ise**
        $ekle\ E \rightarrow \mathcal{E}(O_i, k)$
    $yon_2 \leftarrow \mathcal{Y}\big(O_{i,k}\big)$
    $son\_gozlem\big(O_{i,k}\big) \leftarrow (i, O_i, k, yon_2)$
**döndür** $\mathcal{E}$

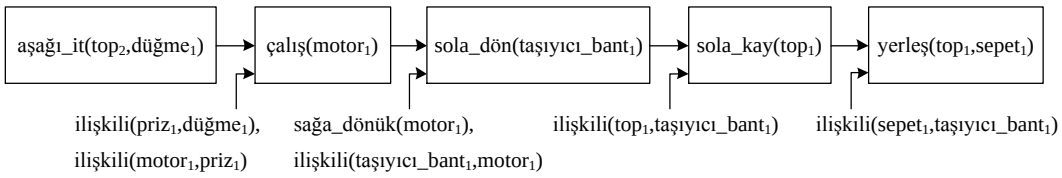
## 7.7 Planlama Uygulaması

Öğrenme çıktılarının planlama için uygulanabilirliğini göstermek üzere, Şekil 7.10 (a)'da ilk durumu verilen örnek problem hazırlanmıştır. Bu örnekte amaç, verilen nesneleri uygun şekilde birbirleri ile ilişkilendirerek $top_1$ nesnesinin sepete girmesini sağlayacak $\big(yerleş(top_1, sepet_1)\big)$ zincirleme bir tepki oluşturmaktır. Problemin çözümüne ulaşmak için, şeklin sağındaki parça kutusunda verilen $motor_1$, $priz_1$ ve $top_2$ nesneleri ve *kayış* bağlantısı ile ortamda halihazırda konumlandırılmış olan $taşıyıcı\_bant_1$, $top_1$ ve $sepet_1$'in birbirleri ile etkileşim içinde kullanılması gerekmektedir.

Verilen nesneler için eğitim senaryolarından öğrenilen etkileşimler kullanılarak, hedef durumdan ilk duruma doğru arama (geri durum uzayı arama) yapıldığında Şekil 7.11'de verilen sıralı olaylara ve etkileşim koşullarına ulaşılabilmektedir. *ilişkili* adıyla ifade edilen önkoşullar Şekil 7.10 (b)'deki gibi sağlandığında, bulunan bu sıralı olaylar üzerinden problemin çözülmesi mümkün olmaktadır. Şekilde *ilişkili* adıyla belirtilen bu önkoşullar, sistemde kullanılan çıkarsama yaklaşımına (bilgi tabanlı, uzamsal, zamansal veya uzam-zamansal) bağlı olarak ilgili ilişki adıyla (*sahip*, *takılı*, *teğet* vb.) ifade edilmektedir.


**(a)** **(b)**

**Şekil 7.10:** (a) Örnek bir TIM problemi ve (b) problemin çözümü.




**Şekil 7.11:** Problemin çözümü için gerçekleşmesi gereken olaylar ve koşulları.

Ortamda oyuncu tarafından uygulanabilen eylemlerin (*konumlandır*, *döndür*, *kayışla_bağla* ve *iple_bağla*) tanımları sisteme verildiğinde, problemin çözümü için gereken koşulların ($priz_1$ ve $motor_1$ ($priz_1$'e takılı olacak şekilde) ortamda konumlandırılmalı, $motor_1$ sağa döndürülmeli ve $taşıyıcı\_bant_1$ $motor_1$'e kayışla bağlanmalı) sağlanmasına yönelik bir plan üretilebilmektedir. Etkileşimlerin öğrenilmesi sırasında, nesnelerin fizik modelleri ve hareketi hesaba katılmadığından $top_2$'nin $düğme_1$'in yukarısında konumlandırılması gerektiği belirlenememektedir.

# 8. BAŞARIM ANALİZİ VE DENEYSEL SONUÇLAR

## 8.1 Uzamsal ve Zamansal Çıkarsama Yöntemlerinin Genel Analizi

Uzamsal çıkarsama ile yakın mesafeden birbirleri ile etkileşim kuran nesneler arasındaki ilişkiler ve parça-bütün ilişkileri belirlenebilmektedir. Bununla birlikte, çıkarsama için sadece uzamsal yerellikten faydalanılmasının bazı sakıncaları bulunmaktadır. Öncelikle, birbirinden uzak olmasına rağmen etkileşim kurabilen nesnelerin arasındaki ilişki (örneğin, kayış veya iple bağlı olan nesneler, uzaktan kumanda ile kontrol edilen dinamit vs.) bu yaklaşımla bulunamaz. Ayrıca, birbirleri ile yakın konumlarda olmasına rağmen aralarında herhangi bir ilişki bulunmayan nesneler de (örneğin, Şekil 7.1'deki motorlar) uzamsal çıkarsamaya dayalı yaklaşımda ilişkili olarak nitelendirilmektedir.

Olayların zamansal bilgilerinden faydalanılarak birbirini uzak mesafelerden etkileyebilen nesneler arasındaki ilişkilerin (örneğin, dinamit ve uzaktan kumanda arasındaki ilişki) modellenmesi mümkündür. Bununla birlikte, sadece zamansal bilgilere göre çıkarım yapmanın da bir takım dezavantajları bulunmaktadır. Bu yöntemdeki en büyük problem, birbirini takip eden iki olayın argümanlarındaki nesnelerden hangilerinin birbirini etkilediğinin belirlenememesidir. Örneğin, Şekil 7.7'deki $O_1 = aşağı\_it(top_1, düğme_1)$ ve $O_2 = çalış(motor_1)$ olayları arasındaki zamansal ilişkide $motor_1$'in çalışmasını başlatan nesnenin $top_1$ ya da $düğme_1$ olduğuna karar vermek mümkün olmamaktadır. Ayrıca birden fazla önkoşulu bulunan *ekmek_kızart* gibi olaylar modellenirken de zamansal yaklaşım başarılı sonuçlar üretememektedir (ekmeğin kızartılması için ekmek_kızartıcı elektriğe bağlı olmalı ve düğmesine basılmış olmalı).

Uzamsal ve zamansal yaklaşımların iyi yanlarını bir araya getiren uzam-zamansal yaklaşımda, zamansal çıkarsamadaki argüman seçim problemi en aza indirilmekte ve birden fazla önkoşula sahip etkileşimlerin (örn., *ekmek_kızart*) öğrenilmesi mümkün olmaktadır. Ayrıca, eş zamanlı bağımsız etkileşim zincirlerinin olduğu eğitim senaryolarında da uzam-zamansal çıkarsama ile başarılı sonuçlar alınabilmektedir.

## 8.2 Öğrenme Deneyleri

Önerilen sistemi sınamak üzere TIM oyununda 40 farklı nesne türü arasındaki çeşitli etkileşimleri barındıran 25 örnek eğitim senaryosu hazırlanmıştır. Sunulan yöntemlere ilişkin sonuçlar dört farklı başarım ölçütü (*doğruluk*, *kesinlik*, *anma* ve *F-ölçütü*) göz önüne alınarak analiz edilmiştir. Yapılan analizde eğitim senaryolarında gerçekte olması gereken olay önkoşulları ve koşullu etkilerle sistem tarafından üretilen sonuçlar birbirleri ile karşılaştırılmıştır. Her bir olayın her bir argümanında bulunan tüm $N_i$ nesneleri için:

- nesne üzerindeki bir etki doğru olarak tespit edildiğinde *gerçek pozitif*,
- olmayan bir etkileşim öğrenilmediğinde *gerçek negatif*,
- hatalı bir etkileşim bulunduğunda *yanlış kabul*,
- ve bulunması gereken bir etki tespit edilemediğinde *yanlış ret* olarak sınıflandırılmıştır.

25 eğitim senaryosundaki nesneler arasında gerçekleşen 175 etkileşim üzerinde bilgi tabanlı yaklaşım, uzamsal çıkarsama, zamansal çıkarsama ve uzam-zamansal çıkarsamayla öğrenilen sonuçlara ilişkin *gerçek pozitif* ($n_{gp}$), *gerçek negatif* ($n_{gn}$), *yanlış kabul* ($n_{yk}$) ve *yanlış ret* ($n_{yr}$) sayıları Çizelge 8.1'de verilmiştir.

**Çizelge 8.1:** Deneylerde elde edilen değerler.

| | **Gerçek Pozitif** | **Gerçek Negatif** | **Yanlış Kabul** | **Yanlış Ret** |
|---|---|---|---|---|
| **Bilgi tabanlı yaklaşım** | 116 | 56 | 1 | 2 |
| **Uzamsal çıkarsama** | 71 | 50 | 25 | 29 |
| **Zamansal çıkarsama** | 86 | 46 | 43 | 0 |
| **Uzam-zamansal çıkarsama** | 114 | 50 | 11 | 0 |

Bu değerler kullanılarak *doğruluk*, *kesinlik*, *anma* ve *F-ölçütü* aşağıdaki formüllerle (8.1) hesaplanabilir.

$$Kesinlik = \frac{n_{gp}}{n_{gp} + n_{yk}} \qquad Doğruluk = \frac{n_{gp} + n_{gn}}{n_{gp} + n_{gn} + n_{yk} + n_{yr}}$$

$$Anma = \frac{n_{gp}}{n_{gp} + n_{yr}} \qquad F - Ölçütü = 2 * \frac{Kesinlik * Anma}{Kesinlik + Anma}$$

**(8.1)**

Deneylerde elde edilen sonuçlar Çizelge 8.2'de gösterilmiştir. Beklendiği üzere bilgi tabanlı yaklaşım diğer yaklaşımlardan daha iyi sonuçlar üretmiştir. Fakat bazı senaryolarda doğrudan gözlemlenemeyen ilişkiler (örneğin, uzaktan kumanda ile kontrol ettiği dinamit arasındaki bağlantı) olması ve eş zamanlı etkileşimler barındıran senaryolardan birinde ürettiği *yanlış kabul* nedeniyle %1.71 oranında hata gözlemlenmiştir.

Uzamsal çıkarsamada birbiri ile ilişkisi olmayan nesnelerin yakın konumlarda bulunduğu senaryolarda oluşan *yanlış kabuller* ve birbirini uzak mesafeden etkileyebilen nesneler nedeniyle oluşan *yanlış retler* dolayısıyla diğer yaklaşımlara göre daha kötü sonuçlar elde edilmiştir.

Zamansal çıkarsamada çoğunlukla ilişkili olaylar arasında argüman seçimindeki hatalardan kaynaklanan *yanlış kabuller* gözlenmiştir. Ayrıca, eş zamanlı bağımsız olay zincirleri barındıran eğitim senaryolarda zamansal çıkarsama ile etkileşimler istenilen seviyede öğrenilememiştir.

Uzam-zamansal çıkarsama, ilişkili argüman seçiminde daha başarılı olduğundan ve birden fazla önkoşula sahip olan olayları da tespit edebildiğinden, zamansal çıkarsamaya göre iyi sonuçlar üretmiştir. Ayrıca, birbirinden bağımsız olay zincirlerinin olduğu Şekil 7.1'dekine benzer senaryolardaki öğrenme başarımı da yüksektir.

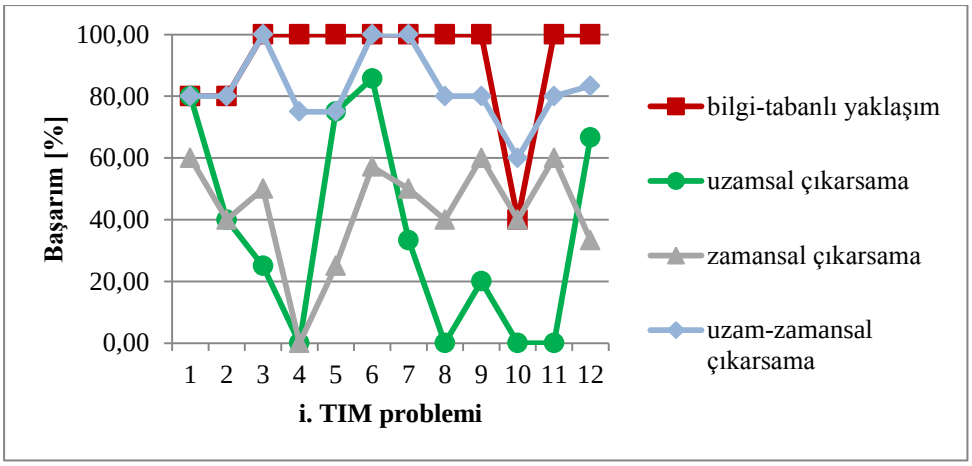
*Doğruluk* ve *F-ölçütü* değerleri göz önüne alındığında uzam-zamansal yaklaşımın bilgi tabanlı yaklaşıma yakın sonuçlar ürettiği gözlenmektedir. Bu sonuçların ışığında, ortamdaki nesneler arası etkileşimlerin parça ilişkilerine dair bilgi kullanılmaksızın sadece uzamsal ve zamansal bilgilere dayanarak önemli ölçüde öğrenilebildiği görülmektedir. Çizelge 8.2'deki sonuçların yanı sıra, deneylerde $\mathcal{Y}$ yönelimlerinin kullanımının öğrenme başarımına katkısı %23.75 olarak gözlenmiştir.

**Çizelge 8.2:** Başarım ölçütleri kullanılarak sonuçların analizi.

| | **Doğruluk** | **Kesinlik** | **Anma** | **F-ölçütü** |
|---|---|---|---|---|
| **Bilgi tabanlı yaklaşım** | % 98.29 | % 99.15 | % 98.31 | % 98.73 |
| **Uzamsal çıkarsama** | % 69.14 | % 73.96 | % 71.00 | % 72.45 |
| **Zamansal çıkarsama** | % 75.43 | % 66.67 | % 100.00 | % 80.00 |
| **Uzam-zamansal çıkarsama** | % 93.71 | % 91.20 | % 100.00 | % 95.40 |

### 8.3 Planlama Deneyleri

Sistemin öğrenme başarımının analiz edilmesinden sonra, TIM oyununda 12 farklı problem hazırlanarak, eğitim senaryolarında öğrenilen etkileşimlerin çözüme yönelik plan üretilirken kullanımı sınanmıştır. Sistemin planlama başarımını ölçmek için, hedef duruma ulaşılırken gerçekleşmesi gereken olayların önkoşullarını sağlamak üzere ortamda uygulanabilen *konumlandır*, *döndür*, *kayışla_bağla* ve *iple_bağla* eylemlerinin hangi oranda doğru olarak seçildiği analiz edilmiştir. Elde edilen sonuçlar Şekil 8.1'de görülmektedir. Öğrenme başarım analizindeki sonuçlara benzer şekilde, bilgi tabanlı yaklaşımın çoğu problemde en iyi sonucu verdiği görülmektedir. Yalnız, eğitim aşamasında öğrenilemeyen dinamitle uzaktan kumanda arasındaki ilişkiyi barındıran 10. problem üzerindeki başarım düşük çıkmıştır. Bazı problemlerde plan yapılırken, uzamsal çıkarsama zamansal çıkarsamaya göre daha başarılı iken, bazı problemlerde zamansal çıkarsama daha iyi sonuçlar doğurmaktadır. Çoğu problemde tek başına uzamsal bilgi ya da zamansal bilgi kullanılarak elde edilen sonuçlar planlama için yetersiz kalmaktadır. Uzam-zamansal yaklaşım, genel olarak bilgi tabanlı yaklaşıma yakın doğrulukta planlar üretmektedir. Bilgi tabanlı yaklaşımla ortalama plan üretme başarım oranı % 91.80 iken, uzam-zamansal yaklaşım için bu oran %83.61'dir.




**Şekil 8.1:** Farklı çıkarsama yaklaşımlarının 12 problemde planlama başarımı.

# 9. SONUÇLAR VE İLERİ ÇALIŞMALAR

Bu tez çalışmasında, ortamda bulunan nesneler arasındaki etkileşimleri metin-tabanlı eğitim senaryoları üzerinden öğrenebilen bir sistem tasarlanmıştır. Sisteme girdi olarak verilen eğitim senaryolarında, ortamdaki nesneler üzerinde gerçekleşen olayların sıralı dizisi, farklı yönelimlerde bulunabilen nesnelerin yönelim bilgileri ve nesneler arasında insan seviyesinde (*human-level*) doğrudan gözlemlenebilen ilişkileri tutan bir bilgi tabanı bulunmaktadır. Öğrenme sürecinde, bu senaryolar dışında nesneler ve aralarındaki etkileşimler konusunda anlamsal (*semantic*) bilgi kullanılmamaktadır. Nesneler arasındaki ilişkileri barındıran bir bilgi tabanı sisteme sağlanmadığında, nesnelerin uzamsal bilgileri ile olayların zamansal bilgileri üzerinden çıkarsama yapılarak bilgi tabanı oluşturulmaktadır.

Geliştirilen sistemin ilk aşamasında ortamdaki nesneler olaylar üzerinden gruplara ayrılmakta ve her bir nesne grubunu (ve farklı yönelimlerini) modelleyen sonlu durum makineleri oluşturulmaktadır. Sonlu durum makineleri oluşturulduktan sonra, sistemin ikinci aşamasında nesneler arasındaki etkileşimler, farklı nesne gruplarının sonlu durum makineleri arasında koşullu bağlantılar oluşturularak öğrenilmektedir.

Sistem üzerinde yapılan öğrenme ve planlama başarım analizleri, nesneler arası ilişkileri içeren bilgi tabanlı yaklaşımla nesnelerin birbirleri üzerindeki etkilerinin benzer problemler verildiğinde çözüm bulunabilecek seviyede öğrenilebileceğini ortaya koymaktadır. Yapılan analizler ayrıca, ortamdaki nesnelere ilişkin uzamsal bilgiler ile nesneler üzerinde gerçekleşen olayların zaman bilgileri uzam-zamansal bir yaklaşımda birlikte kullanıldığında bilgi tabanlı yaklaşıma yakın seviyede sonuçlar elde edilebileceğini göstermektedir. Uzamsal ve zamansal bilgiler kullanılarak öğrenmenin mümkün olabilmesi, doğrudan gözlemlenebilen ilişkilerin bir uzman tarafından sağlanmasından daha az maliyetli olduğundan büyük önem taşımaktadır.

İleri çalışmalarda, bu aşamada metin-tabanlı olarak verilen girdilerin bilgisayarla görü teknikleri ile otomatik bir şekilde elde edilmesi ve bu çalışmada önerilen

öğrenme yaklaşımının gerçek robotlardaki planlama uygulamalarında nesne kullanımı için nesne işlevlerinin öğrenilmesi problemlerine uyarlanması planlanmaktadır.

## KAYNAKLAR


**The Incredible Machine 3 [PC Software].** (1995). United States: Dynamix Inc.

**Allen, J. F.** (1983). Maintaining Knowledge about Temporal Intervals. *Communications of the ACM*, 832-843.

**Amir, E., & Chang, A.** (2008). Learning Partially Observable Deterministic Action Models. *Journal of Artificial Intelligence Research (JAIR)*, 349-402.

**Blum, A. L., & Furst, M. L.** (1995). Fast Planning Through Planning Graph Analysis. *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI'95)* (s. 1636-1642). Montreal, Quebec, Canada.

**Brunelli, R.** (2009). *Template Matching Techniques in Computer Vision: Theory and Practice*. Wiley.

**Cresswell, S. N., & Gregory, P.** (2011). Generalised Domain Model Acquisition from Action Traces. *Proceedings of the 21st International Conference on Automated Planning and Scheduling (ICAPS'11)* (s. 42-49). Freiburg, Germany: AAAI Press.

**Cresswell, S. N., McCluskey, T. L., & West, M. M.** (2009). Acquisition of Object-Centred Domain Models from Planning Examples. *Proceedings of the 19th International Conference on Automated Planning and Scheduling (ICAPS'09)* (s. 338-341). Thessaloniki, Greece: AAAI Press.

**Cresswell, S. N., McCluskey, T. L., & West, M. M.** (in press). Acquiring Planning Domain Models Using LOCM. *The Knowledge Engineering Review*.

**Egenhofer, M., & Herring, J.** (1991). *Categorizing Binary Topological Relationships Between Regions, Lines, and Points in Geographic Databases.* Technical Report, Dept. of Surveying Engineering, University of Maine.

**Fikes, R. E., & Nilsson, N. J.** (1971). STRIPS: A New Approach to the Application of Theorem Proving to Problem Solving. *Proceedings of the 2nd International Joint Conference on Artificial Intelligence (IJCAI'71)* (s. 608-620). London, England: Morgan Kaufmann Publishers Inc.

**Frank, A. U.** (1991). Qualitative Spatial Reasoning about Cardinal Directions. *Proceedings of the International Symposium on Computer-Assisted Cartography (Auto-Carto 10)*, (s. 148-167). Baltimore, Maryland, USA.

**Gil, Y.** (1994). Learning by Experimentation: Incremental Refinement of Incomplete Planning Domains. *Proceedings of the 11th International Conference on Machine Learning (ICML'94)* (s. 87-95). New Brunswick, NJ, USA: Morgan Kaufmann.

**Horn, A.** (1951). On Sentences Which are True of Direct Unions of Algebras. *Journal of Symbolic Logic*, 14-21.

**Kalisch, M., & König, S.** (2005). *Comparison of STRIPS, ADL and PDDL*. Nisan 2012 tarihinde http://www.cogsys.wiai.uni-bamberg.de/teaching/ws0405/s_planning/slides/Introduction_AI_Planning_addon.pdf adresinden alındı

**McCluskey, T. L., Cresswell, S. N., Richardson, N. E., & West, M. M.** (2009). Automated Acquisition of Action Knowledge. *Proceedings of the 1st International Conference on Agents and Artificial Intelligence (ICAART'09)* (s. 93-100). Porto, Portugal: INSTICC Press.

**McCluskey, T. L., Richardson, N. E., & Simpson, R. M.** (2002). An Interactive Method for Inducing Operator Descriptions. *Proceedings of the 6th International Conference on Artificial Intelligence Planning Systems (AIPS'02)* (s. 121-130). Toulouse, France: AAAI Press.

**McDermott, D., Ghallab, M., Howe, A., Knoblock, C., Ram, A., Veloso, M.,** et al. (1998). *PDDL - The Planning Domain Definition Language.* Yale Center for Computational Vision and Control.

**Mitra, D.** (2002). Qualitative Reasoning with Arbitrary Angular Directions. *In Proceedings of AAAI'02 Workshop on Spatial and Temporal Reasoning.* Edmonton, Alberta, Canada: AAAI Press.

**Nau, D. S.** (2009). *Lecture slides for Automated Planning: Theory and Practice.* Nisan 2012 tarihinde http://www.cs.umd.edu/~nau/planning/slides/ adresinden alındı

**Pednault, E. P.** (1989). ADL: Exploring the Middle Ground Between STRIPS and the Situation Calculus. *Proceedings of the 1st International Conference on Principles of Knowledge Representation and Reasoning ( KR'89)* (s. 324-332). Toronto, Canada: Morgan Kaufmann Publishers Inc.

**Pujari, A. K., Kumari, G. V., & Sattar, A.** (1999). INDU: An Interval and Duration Network. *Proceedings of the 12th Australian Joint Conference on Artificial Intelligence: Advanced Topics in Artificial Intelligence (AI'99)* (s. 291-303). Sydney, Australia: Springer-Verlag.

**Randell, D. A., Cui, Z., & Cohn, A. G.** (1992). A Spatial Logic based on Regions and Connection. *Proceedings of the 3rd International Conference on Principles of Knowledge Representation and Reasoning ( KR'92)* (s. 165-176). Cambridge, Massachusetts, USA: Morgan Kaufmann.

**Richardson, M., & Domingos, P.** (2006). Markov Logic Networks. *Machine Learning*, 107-136.

**Rube Goldberg Inc.** (2012). *Gallery*. Nisan 2012 tarihinde The Official Rube Goldberg Website: http://www.rubegoldberg.com/?page=gallery adresinden alındı

**Russell, S., & Norvig, P.** (2002). *Artificial Intelligence: A Modern Approach (2nd Edition).* Prentice Hall.

**Shahaf, D., & Amir, E.** (2006). Learning Partially Observable Action Schemas. *Proceedings of the 21st National Conference on Artificial Intelligence (AAAI'06)* (s. 913-919). Boston, Massachusetts, USA: AAAI Press.

**Shahaf, D., Chang, A., & Amir, E.** (2006). Learning Partially Observable Action Models: Efficient Algorithms. *Proceedings of the 21st National Conference on Artificial intelligence (AAAI'06)* (s. 920-926). Boston, Massachusetts, USA: AAAI Press.

**SRI International.** (2012). *Shakey*. Nisan 2012 tarihinde SRI International's Artificial Intelligence Center: http://www.ai.sri.com/shakey/ adresinden alındı

**Vilain, M. B.** (1982). A System for Reasoning About Time. *Proceedings of the 2nd National Conference on Artificial Intelligence (AAAI'82)* (s. 197–201). Pittsburgh, PA, USA: AAAI Press.

**Vilain, M., Kautz, H., & van Beek, P.** (1989). Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report. *Readings in Qualitative Reasoning about Physical Systems* (s. 373-381). Morgan Kaufmann Publishers Inc.

**Walsh, T. J., & Littman, M. L.** (2008). Efficient Learning of Action Schemas and Web-Service Descriptions. *Proceedings of the 23rd National Conference on Artificial Intelligence (AAAI'08)* (s. 714-719). Chicago, Illinois, USA: AAAI Press.

**Wang, X.** (1995). Learning by Observation and Practice: An Incremental Approach for Planning Operator Acquisition. *Proceedings of the 12th International Conference on Machine Learning (ICML'95)* (s. 549-557). Tahoe City, California, USA: Morgan Kaufmann.

**Wood, J.** (2008). Minimum Bounding Rectangle. S. Shekhar, & H. Xiong içinde, *Encyclopedia of GIS* (s. 660-661). Springer.

**Wu, K., Yang, Q., & Jiang, Y.** (2007). ARMS: An Automatic Knowledge Engineering Tool for Learning Action Models for AI Planning. *The Knowledge Engineering Review*, 135-152.

**Yang, Q., Wu, K., & Jiang, Y.** (2005). Learning Action Models from Plan Examples with Incomplete Knowledge. *Proceedings of the 15th International Conference on Automated Planning and Scheduling (ICAPS'05)* (s. 241-250). Monterey, California, USA: AAAI Press.

**Zhuo, H. H., Yang, Q., Hu, D. H., & Li, L.** (2010). Learning Complex Action Models with Quantifiers and Logical Implications. *Artificial Intelligence*, 1540-1569.

## ÖZGEÇMİŞ

| | |
|---|---|
| **Ad Soyad:** | Mustafa Ersen |
| **Doğum Yeri ve Tarihi:** | Burdur, 13 Eylül 1985 |
| **E-Posta:** | ersenm@itu.edu.tr |
| **Lisans:** | İstanbul Teknik Üniversitesi<br>Bilgisayar Mühendisliği Bölümü (2009) |
| **Mesleki Deneyim:** | Araştırma Görevlisi (2010 - )<br>İstanbul Teknik Üniversitesi<br>Bilgisayar Mühendisliği Bölümü |

### TEZDEN TÜRETİLEN YAYINLAR/SUNUMLAR

- **Ersen, M., and Sariel-Talay, S.,** 2012: Learning Interactions Among Objects Through Spatio-Temporal Reasoning. *AAAI-12 Workshop on Problem Solving using Classical Planners (CP4PS'12)*, Toronto, Ontario, Canada.

- **Ersen, M., and Sariel-Talay, S.,** 2012: Learning Interactions Among Objects, Tools and Machines for Planning. *The Seventeenth IEEE Symposium on Computers and Communication (ISCC'12)*, Cappadocia, Turkey.

- **Ersen, M., ve Sariel-Talay, S.,** 2012: Uzam-zamansal Çıkarsamayla Nesneler Arası Etkileşimlerin Öğrenilmesi. *20. IEEE Sinyal İşleme ve İletişim Uygulamaları Kurultayı (SİU'12)*, Ölüdeniz, Fethiye, Muğla.