master ISICG/TIM

Greek Language Object Representation Scene System (GLOReSS)

A text interface for designing X3D objects and X3D scenes

MASTER in INFORMATICS, IMAGE SYNTHESIS and GRAPHICS DESING

George Krikos
Supervised by Nikitas N. Karanikolas

1 INDRODUCTION

2 SYSTEM GENERAL OVERVIEW

3 DEEP SYSTEM DESIGN

4 CASE STYDIES

5 CONCLUTIONS

INTRODUCTION

Project description

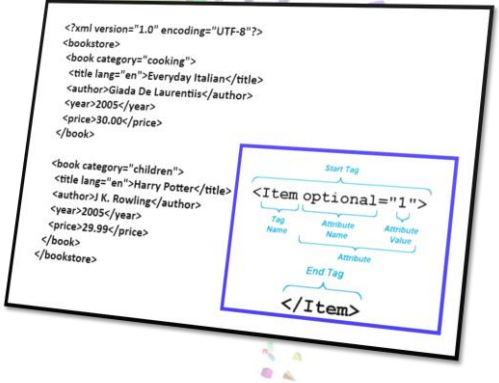
- New tool for creating 3D scenes (text to scene system)
- Text input description in natural Greek language
- Based on basic 3D shapes
- Automatic generating scenes system
- System consists an interface between the text description and the final produced 3D scene
- Final output is a graphical representation to a simple html
- Fundamental actions included
- Opportunity to reuse created objects

INTRODUCTION

Technologies (1/2)

- Extensible 3D (X3D)
  - It is a file format for representing 3D scenes
  - Successor of Virtual Reality Modeling Language (VRML)
  - Multiplatform support
  - X3D inline as XML in a HTML page

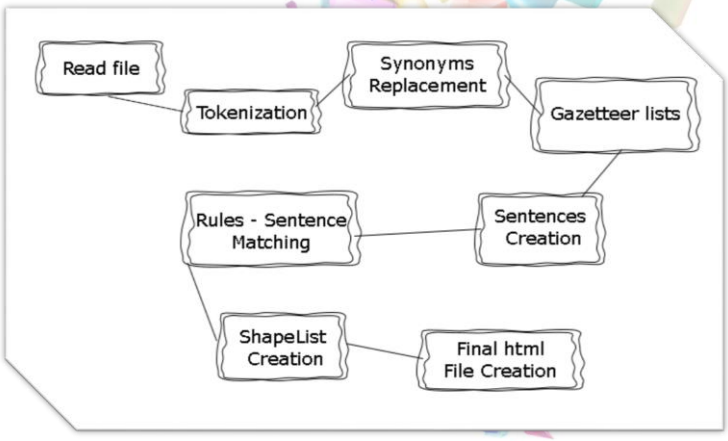
- eXtensible Markup Language (XML)
  - Encode data both human and machine can understand
  - Set of rules to describe data (content of files)
  - Emphasize to simplicity and usability across the internet
  - User is totally free to create his own tags



INTRODUCTION

Technologies (2/2)

- **Natural Language Processing (NLP)**
  - Method to analyze large amount of data
  - Data expressed by spoken or written natural language.
  - Machine learning approach and Rule based approach
  - Information extraction (IE) is a subfield of NLP. Scopes to extract specific information from text sources (text mining)
  - Other subfields are speech recognition, natural language understanding and generation, automatic summarization, etc.



INTRODUCTION

Previous Work on Text to Scene Systems

Natural Language driven Image Generation (NALIG)

- NALIG is an early text to scene system
- Created for experimental purposes at the University of Genoa (1984)
- Designed static 2D scenes from simple phrases in Italian language

WordsEye

- WordsEye (2001) is a system to convert simple English text into 3D graphical scenes
- It features a large database of 3D objects
- Image is generated from a description of how some objects are placed in a scene
- The description consists of spatial relations
- Rules are applied to resolve conflicts and add implicit constraints

Cognitive Vision System (CogViSys)

- CogViSys (2001) visualizing descriptions of simple vehicle behavior at intersections
- The basic idea was of generating texts from a sequence of video images

SYSTEM GENERAL OVERVIEW
System implementation

- Java programing language
- Developed into Eclipse IDE Neon (tested also at Oxygen)
- WindowBuilder Java plug-in to create Java GUI applications (forms)
- x3dom files included (x3dom.js, x3dom.css)
- Uses text files as configuration
- Input file are a text file
- Output file is an html page

- Input file
  - A sentence divided into 3 different parts (X - V - Y). main object, verb and all features
  - The verbs are in imperative format
  - Any order for 3 parts

Example:

"ένα κόκκινο κύβο κατασκεύασε μπροστά από την κίτρινη σφαίρα"

or

"κατασκεύασε ένα κόκκινο κύβο μπροστά από την κίτρινη σφαίρα"

or

"κατασκεύασε μπροστά από την κίτρινη σφαίρα ένα κόκκινο κύβο"

or

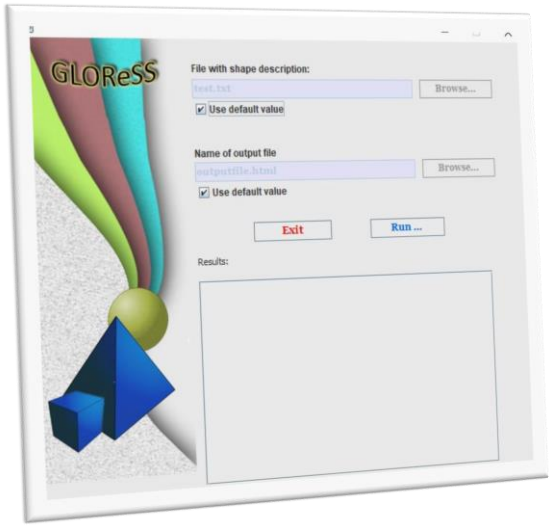
"μπροστά από την κίτρινη σφαίρα κατασκεύασε ένα κόκκινο κύβο".

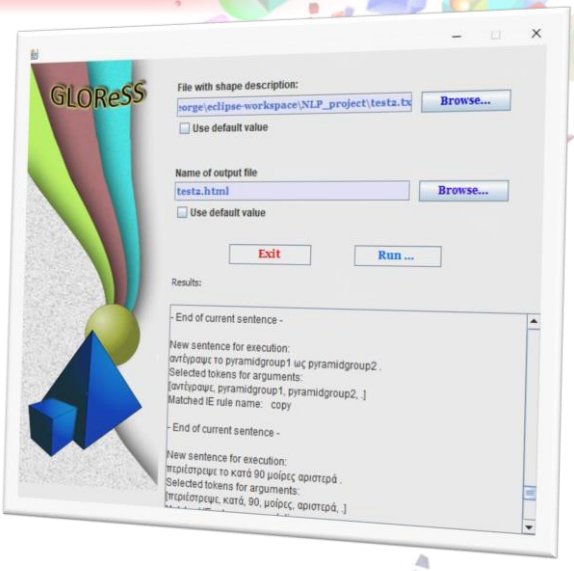
SYSTEM GENERAL OVERVIEW
User Interface (1/2)

- User interface simple to handle
- Offers text description input file selection as well as final output file name.
- Browse window for files selection
- Offers default values for both files
  - text.txt
  - outputfile.html
- Text field area for messaging interaction and step watching
- Execute and exit buttons.

SYSTEM GENERAL OVERVIEW

User Interface (2/2)





SYSTEM GENERAL OVERVIEW

Files of System

- Gazeetter lists
  - Colors
  - Shapes
  - Verbs
  - Keywords
  - Directions
- Synonyms file
- Output file
- Rule files
  - FL rules
  - IE rules

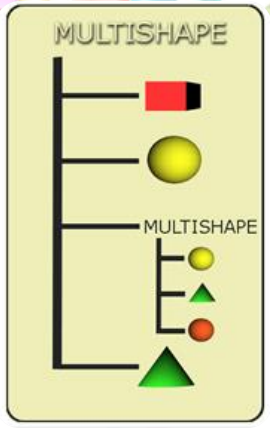
χρωμάτισε {βάψε, άλλαξε το χρώμα}
περιέστρεψε {στρίψε, γύρισε}
μεγέθυνε {αύξησε τις διαστάσεις, μεγάλ
ονόμασε {δώσε το όνομα, ονομάτισε}
ομαδοποίησε {κάνε ομάδα, συνένωσε}
κόκκινο {κόκκινη, κόκκινου, κόκκινης, κό
πράσινο {πράσινη, πράσινου, πράσινη
κίτρινο {κίτρινη, κίτρινου, κίτρινης, κίτριν
σημείο {σημεία, θέση}
ένα {μία, μια, έναν}
του {της}
το {την, τον, τη}
διάστασης {μέγεθος, με διαστάσεις, μεγέθους, με διάσταση, διαστάσεων}

SYSTEM GENERAL OVERVIEW

Shapes and actions

- **Basic shape utilize**
  - {Cube, Cuboid, Cone, Cylinder, Pyramid, Squared Pyramid, Sphere}

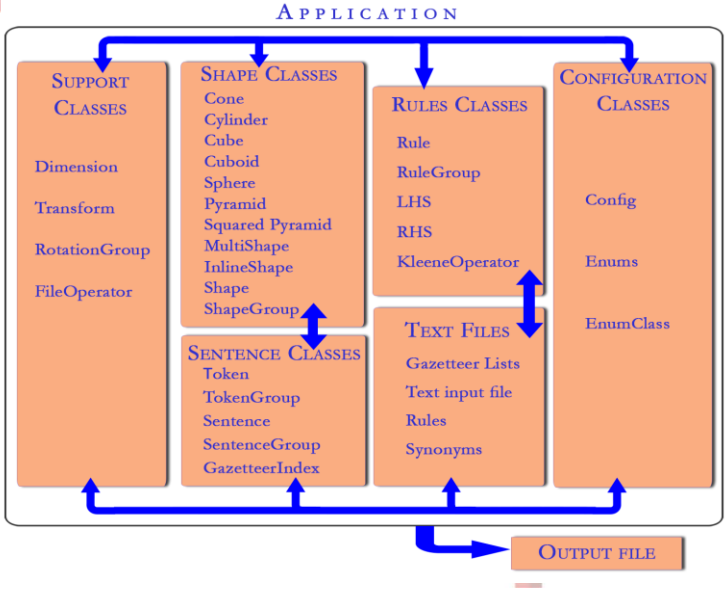
- **Two special shape classes**
  - MultiShape (group of shapes)
  - InlineShape (external file as library)
- **Fundamental actions and more**
  - {create, move, rotate, scale, copy, clone, inline, group}



SYSTEM GENERAL OVERVIEW

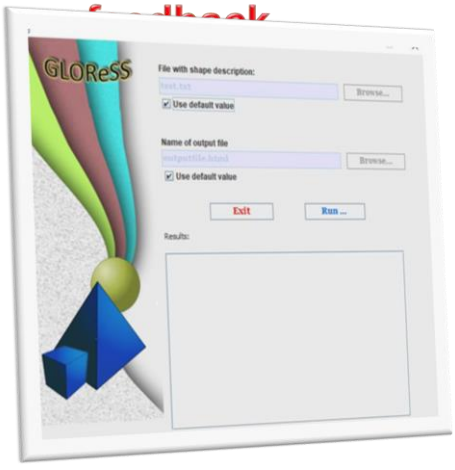
Classes interaction model

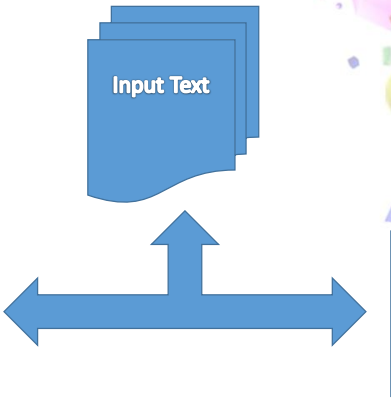
- ✓ TEXT FILES can characterized as configuration files
- ✓ SUPPORT CLASSES help system to handle the text files (open, close, read, write, etc.) and parameters like dimensions and rotations
- ✓ Shape class consist the super class for all other classes into SHAPE CLASSES
- ✓ RULES CLASSES manage the rules
- ✓ CONFIGURATION sets program variables
- ✓ For tokenization, sentences splitting and tokens category definition is responsible the SENTENCE



SYSTEM GENERAL OVERVIEW

Error Messaging





DEEP SYSTEM DESIGN

Input file issues

- User describes the scene into a text file.
- Input file contains simple sentences for designing a scene based on basic shapes.
- Transformations and other actions are available.
- An verb (action) required for sentence execution.
- User will get the desire final result after successive corrections to the input file.
- Some actions contain default parameters values, while others should defined by user.

DEEP SYSTEM DESIGN

Input file issues

- Action **"κατασκεύασε"** creates a shape
  - Required argument is the type of the shape (etc. cube).
  - Optional arguments {color, position, name}.
  - Position can be a direct or an indirect point.
  - Negative numbers are accepted.
- **"μετακίνησε"** moves a shape
  - Support direct or indirect position. Also moves towards a direction. (e.g. Μετακίνησε ***πάνω*** κατά 3…)
  - A direction word is required for every indirect move. (*αριστερά, δεξιά, μπροστά, πίσω, κάτω, πάνω*)
  - Direction words always followed by a number or a shape or both.
  - Negatives numbers are allowed but will move the shape to the opposite direction.

DEEP SYSTEM DESIGN

Input file issues

- Verb **"περιέστρεψε"** rotates a shape.
  - Must contain a direction word. (*αριστερά, δεξιά, πίσω, μπροστά, πλάγια αριστερά, πλάγια δεξιά*)
  - MultiShape shape center point computed every time a new shape grouped.
  - Negative numbers results an inverse rotation.
- To colorize a shape **"χρωμάτισε"** is applied.
  - Coloring is possible for all the basic shapes.
  - Copy shape support coloring.
  - Clone shape, Multishape and IncludeShape is not possible to change their colors.
  - Clone shape color totally affected by the original shape.

DEEP SYSTEM DESIGN

Input file issues

- Naming a shape applied by **"ονόμασε"**
  - It is a user common task because of better organizing and controlling the shapes.
  - Naming is not only applied with this way. (create, group, copy, clone, inline)
  - Grouping and inline actions need a name, else execution stops with an error message
  - The name must enclosed into double quotes. ("*name_of_shape*")
  - Keyword "*ως*" is vital just before the name.
- Action **"μεγέθυνε"** scales a shape
  - The number of scaling must be part of the sentence otherwise an error occurs.
  - Scaling numbers > 1 extends the shape
  - 0 < scaling numbers < 1 downscale the shape.
  - Negatives numbers raise an error.
  - User can scale the shape in one axe (x, y or z). {*μεγέθυνε κατά 3 φορές στον άξονα X*}
  - Not affects clone and copy shapes.

DEEP SYSTEM DESIGN

Input file issues

- **"αντέγραψε"** and **"κλωνοποίησε"** actions create a new similar shape
  - Copy action uses all the attribute values of the original shape, except the name.
  - Whether the original shape change its color or move to a group shape will not affect the copy one.
  - From the time the copy shape is created is totally independent by the source shape.
  - Cloning a shape use the HTML-X3D tags DEF and USE.
  - Coloring and grouping original shape dramatically affect the clone shape.
  - User should be careful when copy and clone into complex sentences with word "και".

    E.g. *«αντέγραψε τον μωβ κώνο αριστερά κατά 3 και πάνω κατά 5 ως "κωνο2".*

    will create two new shapes. Move action should applied into these cases.

DEEP SYSTEM DESIGN

Input file issues

- Action **"ομαδοποίησε"** groups a number of shapes
  - Grouping a set of shapes and/or other groups requires a new name.
  - Ungrouping or reference to a shape of group individually is not possible.
  - Copy, clone and transforms are active choices.

- For insertion of an external shape user can use **"ενσωμάτωσε"**
  - HTML-tag <inline…> used.
  - A new name is essential.
  - User should give the path of external file or current directory is preselected.
  - Copy, clone and transform actions are possible for inline shapes.

DEEP SYSTEM DESIGN

Steps of processing

- Split to tokens (Tokenization)
- Replace synonyms
- Define tokens' fields
  - ***Kind*** {WORD, NUMBER, SPECIAL WORD, LIKEPOINT, NAME, …}
  - ***Category*** {UNIDENTIFIED, MULTIPLIER, OFFSET, ROTATION, SHAPE, DIRECTION, …}
  - ***FL Rule Name***
  - ***Value***
- Get the sentences
- Matching the rules
- Execute a sentence
- Display the result

DEEP SYSTEM DESIGN

A Tokenization (Split to tokens)

Splitting all the text into words, punctuations marks, numbers and some others discrete categories for further processing

Example :
μετακίνησε επίσης τον κίτρινο κώνο δεξιά από τον ….

μετακίνησε, επίσης, τον, κίτρινο, κώνο, δεξιά, από, τον….

B Replace Synonyms

Replace tokens with specific similar words (lemmatization)

Example :
κίτρινο {κίτρινη, κίτρινου, κίτρινης, κίτρινος}

DEEP SYSTEM DESIGN

C Define tokens' fields

Sets the fields of each token based on gazetteer lists and regulation expressions

VALUE	KIND	CATEGORY	FL Name
-μετακίνησε-	|SPECIAL_WORD|--	|verb|--	||
-επίσης-	|WORD|--	|unidentified|--	||
-το-	|WORD|--	|unidentified|--	||
-κίτρινο-	|SPECIAL_WORD|--	|color|--	||
-κώνο-	|SPECIAL_WORD|--	|shape|--	||
-δεξιά-	|SPECIAL_WORD|--	|direction|--	||
-από-	|WORD|--	|unidentified|--	||
-το-	|WORD|--	|unidentified|--	||
		.	
		.	
		.	

KIND is a syntactic attribute (like POS)
CATEGORY is a semantic attribute

DEEP SYSTEM DESIGN

D Get the Sentences

Group tokens to create sentences

Example:
μετακίνησε τον κόκκινο κώνο δεξιά κατά 1.5 και μπροστά κατά 1.

-μετακίνησε-	|SPECIAL_WORD|--	|verb|--	||
-το-	|WORD|--	|unidentified|--	||
-κόκκινο-	|SPECIAL_WORD|--	|color|--	||
-κώνο-	|SPECIAL_WORD|--	|shape|--	||
-δεξιά-	|SPECIAL_WORD|--	|direction|--	||
-κατά-	|SPECIAL_WORD|--	|keyword|--	||
-1.5-	|NUMBER|--	|offset|--	||
-και-	|WORD|--	|unidentified|--	||
-μετακίνησε-	|SPECIAL_WORD|--	|verb|--	|null|
-μπροστά-	|SPECIAL_WORD|--	|direction|--	||
-κατά-	|SPECIAL_WORD|--	|keyword|--	||
-1-	|NUMBER|--	|offset|--	||
-.-	|PUNCTUATION|--	|punctuation|--	||

DEEP SYSTEM DESIGN

E Matching the rules

Matches the FL rules with groups of tokens. Then select the proper IE rule.

Example:

-μετακίνησε-	|SPECIAL_WORD|--	|verb|--	||
-το-	|WORD|--	|unidentified|--	||
-κίτρινο-	|SPECIAL_WORD|--	|color|--	|sourceShape|
-κώνο-	|SPECIAL_WORD|--	|shape|--	|sourceShape|
-δεξιά-	|SPECIAL_WORD|--	|direction|--	|directionShape|
-από-	|WORD|--	|unidentified|--	|targetShape|
-το-	|WORD|--	|unidentified|--	|targetShape|
-μπλε-	|SPECIAL_WORD|--	|color|--	|targetShape|
-κύλινδρο-	|SPECIAL_WORD|--	|shape|--	|targetShape|
-.-	|PUNCTUATION|--	|punctuation|--	||

The remaining tokens after removing the *"unidentified"* ones
[μετακίνησε, κίτρινο, κώνο, δεξιά, μπλε, κύλινδρο, .]

The rules that update the column FL_Name for the phrase "κίτρινο κώνο"

```
FL_Rule: shape
[(category == color && category == shape)
        || (category == shape && category == color)
        || (category == shape && string == το && category == color)
        || (category == shape)]
:

FL_Rule: sourceShape
[(FL_Rule == shape)
|| (FL_Rule == mname)]
:
```

thus, they create the chunk "sourceShape" in the previous page

DEEP SYSTEM DESIGN

F Execute the sentence

Gets the argument values and execute the sentence.

Example:

Arg Name	- Arg Type	-Category	- Arg Value
$shape1	- shape	- sourceShape	- κώνο
$color	- color	- sourceShape	- κίτρινο
$name1	- name	- sourceShape	- null
#p	- point	- mToPoint	- null
#shape2	- shape	- targetShape	- κύλινδρο
#color2	- color	- targetShape	- μπλε
#direction1	- direction	- directionShape	- δεξιά
#offset1	- offset	- moffset	- null
#keyOffset	- keyword	- moffset	- null
#name2	- name	- targetShape	- null

G Display the result

Read the shape list and create the output html file

DEEP SYSTEM DESIGN

Rules issues

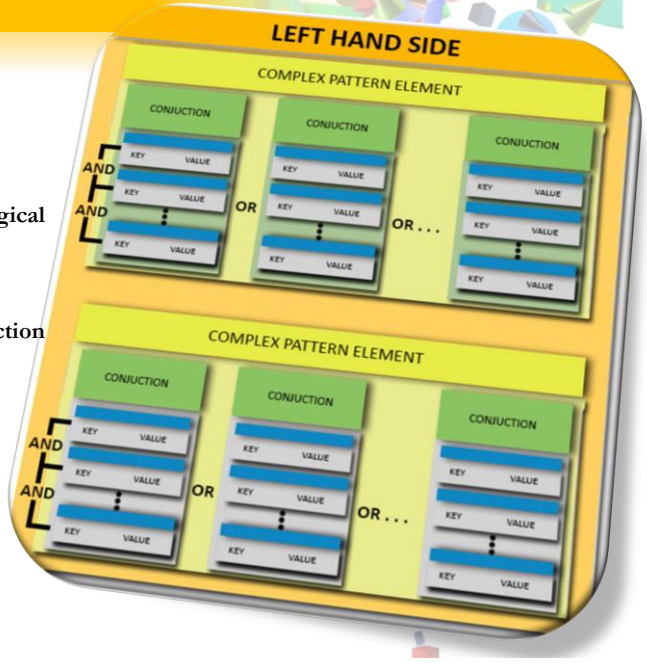
System is a rule based methods extracting data tool. Rules tag the tokens, apply values to parameters and generally is responsible for processing an action.

- Left Hand Side (LHS)
- Right Hand Side (RHS)
  - First Level (FL) Rules
  - Information Extraction (IE) Rules
- Priority of rules

DEEP SYSTEM DESIGN

LEFT HAND SIDE

- Pattern Element (PE): key – value pair .
  (category == number)
- Conjunction list: a list of pattern elements with the logical AND between the elements.
  (string == τον && category == shape).
- Complex Pattern Element (CPE): is a set of conjunction lists between two square brackets.
  [(string == τον) || (string == από)]
- Contains extra sign symbols (*, ?, +)
  ** : zero (0) or more sequential times.*
  ? : can be exist one time or none.
  + : at least one time, maybe more.



DEEP SYSTEM DESIGN

First Level (FL) Rules

- Provide a maximum level of effectiveness to IE rules.
- The FL rule seeks groups of tokens that satisfy the pattern at the left side of the rule.
- Tokens tagged with the name of the FL rule.
- Optional, on RHS, apply changes to the category field of a token.

FL_Rule: mnumber
[(string == κατά && category == number)]
:

FL_Rule: mRotation
[(FL_Rule == mnumber)]
[(string == μοίρες)]
:FL_Rulename --> (number : rotation)

DEEP SYSTEM DESIGN

Information Extraction (IE) Rule

- The IE rules contain the verb of the sentence.
- On the right side of the rule, there are all those attributes that get their values from the sentence and define the shape.
- The IE rules following the FL rules.
- First line specified the name of IE rule
- Two types of arguments
  - starting with the symbol *"$"*. Usually the name, type and color of a shape.
  - Hash symbol *"#"* sets output arguments. Defines the way the main shape will be transformed.
- Directly after the colon (:) there are two word with a period between them.
  - The first details the category of the token and
  - the second defines the name of the FL rule matched for this token.

```
rule: create_Shape
[(string == κατασκεύασε)]
:createShape --> createShape(
$shape1:shape.sourceShape,
$color:color.sourceShape, $name:name.sourceShape,
#p:point.mToPoint, #shape2:shape.targetShape,
#color2:color.targetShape, #offset1:offset.moffset,...
```

DEEP SYSTEM DESIGN

Priority of Rules

Two (2) main steps for rule selection

1. From all rules that matches a region of the sentence starting at some point, the one which matches the longest region is fired.
2. If more than one rules matches then the one defined later in the rule file is fired.

Example:

Assume we have the text "*κατά 3 φορές*".

- second rule would apply, because it matches a longer region of text starting at same point.

Assuming we have only the text "*κατά 3*"

- first and third rules could applied.
- the last rule that matches with the text will be fired.

```
FL_Rule: mnumber
[(string == κατά && category == number)]
:

FL_Rule: mtimes
[(FL_Rule == mnumber)]
[(string == φορές)]
:FL_Rulename --> (number : multiplier)

FL_Rule: moffset
[(FL_Rule == mnumber)]
: FL_Rulechange --> (number : offset)
```

DEEP SYSTEM DESIGN

Assumptions - Acceptances

- The system designed to execute simple proposals
- Separates sentences based on punctuations "." and "," as well as *"και"* word.
- Each sentence refers to one shape at time

  "χρωμάτισε τους δύο κώνους μπλε" is not acceptable

- Actions of each shape applied sequentially without affect more shapes

  Shape (A) moved on the left of a second one (B). Then second change its position. The first shape (A) will not follow the second in order to still stay on its left.

- Similar words handled by a rough approach on lemmatization. A professional lemmatization - steaming tool would offer better effectiveness to the system

DEEP SYSTEM DESIGN

CONSTRAINTS

The description of the shapes and actions must be specific and well-defined within given contexts and principles.

Dimensions of the shape on a specific format
Round brackets (length, height, width).
Each shape uses minimum number of dimensions for definition.

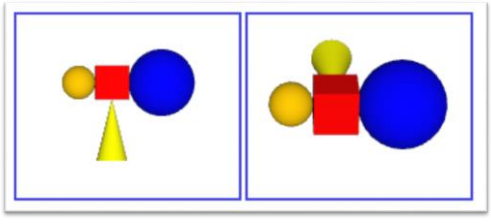
Comma delimiter splits different numbers. Decimal digits separated by dot.
For example "φτιάξε ένα κόκκινο κώνο διαστάσεων (2.5, 3)"

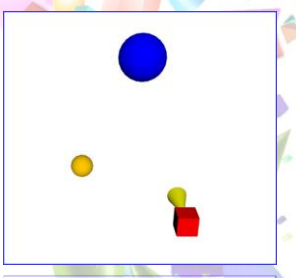
move a shape uses format (X, Y, Z) similar to dimensions definition.

CASE STUDIES

Example (1/3) – Creation and moving

Κατασκεύασε μία πορτοκαλί σφαίρα. Κατασκεύασε ένα κόκκινο κύβο στο σημείο (1, 0, 0). Στη συνέχεια κατασκεύασε ένα κίτρινο κώνο από κάτω του. Επίσης φτιάξε δεξιά από τον κόκκινο κύβο μία μπλε σφαίρα με διάσταση (2).



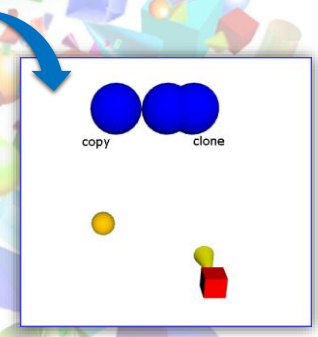


Μετακίνησε τον κόκκινο κύβο και τον κώνο δεξιά κατά 4 και μπροστά κατά 3 και την μπλε σφαίρα πίσω κατά 5.

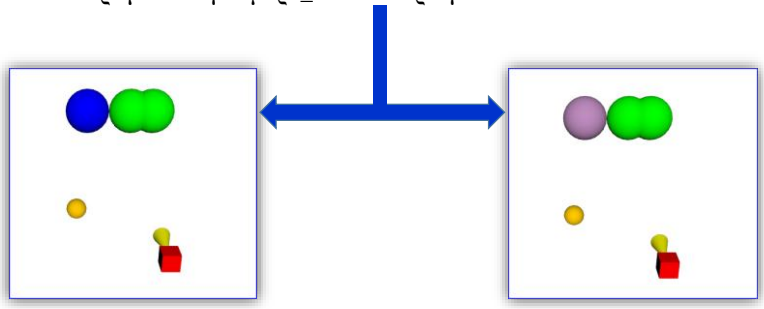
CASE STUDIES

Example (2/3) – copying and cloning

Αντέγραψε την "σφαίρα" ως "σφαίρα_copy" αριστερά της. Κλωνοποίησε την "σφαίρα" ως "σφαίρα_clone" δεξιά κατά 1.



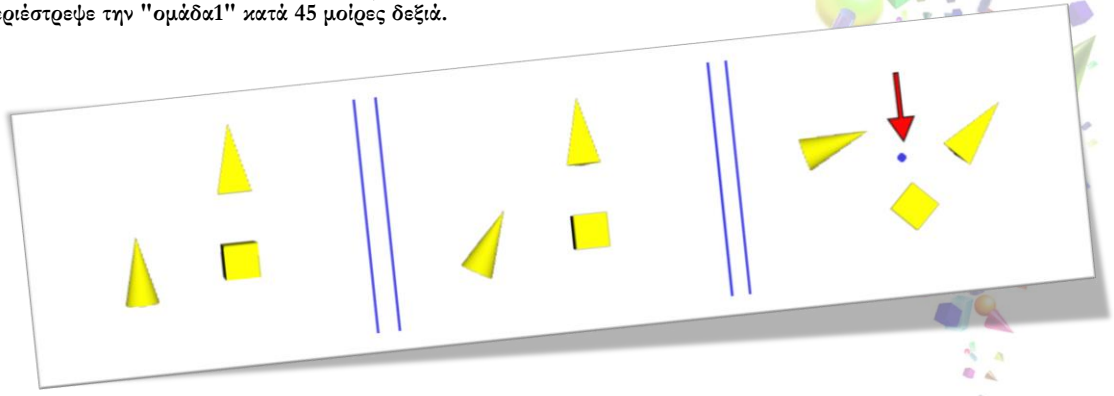
Χρωμάτισε τη "σφαίρα" πράσινη. Χρωμάτισε τη "σφαίρα_copy" μωβ. Χρωμάτισε τη "σφαίρα_clone" κίτρινη.



CASE STUDIES

Example (3/3) – Rotating

Φτιάξε ένα κίτρινο κώνο. Στο σημείο (3,0,0) φτιάξε ένα κίτρινο κύβο και στο σημείο (3,3,0) μία κίτρινη πυραμίδα. Περιέστρεψε τον κώνο κατά 30 μοίρες δεξιά. Ομαδοποίησε τον κώνο και την πυραμίδα και τον κύβο ως "ομάδα1". Περιέστρεψε την "ομάδα1" κατά 45 μοίρες δεξιά.



CASE STUDIES

Problematic examples

- Incomplete description by user.

  *Μετακίνησε τον κώνο **δεξιά** και πίσω κατά 3.*

- confusing sentence for system. Sentence simplicity for best results.

  *Μετακίνησε τον "κώνο1" αριστερά κατά 16 και κάτω κατά 4 και **τον μπλε κύβο** και την κόκκινη πυραμίδα πάνω κατά 5.*

- Words not understandable by system maybe confuse it.

  *Μετακίνησε τον "κώνο1" αριστερά κατά 16 και κάτω κατά 4 **ενώ** τον μπλε κύβο και την κόκκινη πυραμίδα πάνω κατά 8.*

  results following three sentences:

  Μετακίνησε τον "κώνο1" αριστερά κατά 16.

  Μετακίνησε κάτω κατά 4 τον μπλε κύβο.

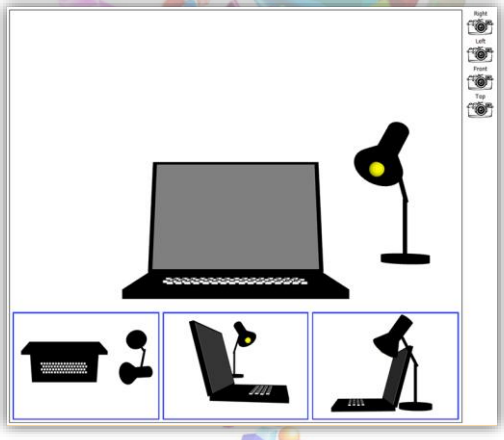
  Μετακίνησε την κόκκινη πυραμίδα πάνω κατά 8.

Conclusion : first sentence will lead to an error but the others will not.

CASE STUDIES

Full example

Φτιάξε ένα μαύρο ορθογώνιο με διαστάσεις (12,8,0.5) στο σημείο (0, 0, 0). Περιέστρεψε το πίσω κατά 15 μοίρες και ονόμασε το ως "screen".
Αντέγραψε το μπροστά από το "screen" ως "base" και περιέστρεψε το κατά 105 μοίρες μπροστά. Μετακίνησε το κάτω από το "screen" και μπροστά κατά 5.
Αντέγραψε το "screen" και μετακίνησε το μπροστά κατά 0.05. ονόμασε το ως "screenIn". Χρωμάτισε το γκρι και μεγέθυνε το κατά 0.95 φορές.
Πάνω από τη "base" φτιάξε ένα άσπρο κύβο με διάσταση (0.35). Μετακίνησε τον κάτω κατά 0.2. Ονόμασε το ως "button1".
Κλωνοποίησε το "button1" αριστερά κατά 0.5 ως "button2", αριστερά κατά 1.0 ως "button3" και αριστερά από το "button3" ως "button4" κατά 0.15.
Ομαδοποίησε το "button1" και το "button2" και το "button3" και το "button4" ως "4buttons". **……**
Ομαδοποίησε τα "8buttons" και "8buttons_2" ως "16buttons".
Κλωνοποίησε το μπροστά κατά 0.6 ως "copy_16buttons". Μετακίνησε το δεξιά κατά 0.3. **……**
Ενσωμάτωσε το αρχείο "lamp.html" ως "2ndItem" στο σημείο (12,0,0). Περιέστρεψε το "2ndItem" κατά 45 μοίρες πλάγια δεξιά.



CONCLUSION & FUTURE WORK

Conclusions

- Users without any requiring knowledge about 3D graphics or specific 3D graphics tools, can create a 3D scene just by describing shapes and their positions into a text file. Just the ability to analyze a complex shape into basic shapes is required.
- The simplicity of using the system help everyone, into their first steps in 3D computer scene design and irritate them to engage more.
- Thus, the final output file is an html file with XML code inline, an easy representative format to any browser and machine

CONCLUSION & FUTURE WORK

Future Work

Shapes and actions
- More shapes could be used as initials shapes as {torus, triangular prism and hexagonal prism, octahedron, dodecahedron, etc}.
- New actions could applied in the future (zoom in/out, between, etc).

Efficiency
- The linguistic problems for more general descriptions are a challenge for an extra work.
- System constructs the scenes according to a specified format with perspective.

Error Handling
- Automatic Error Correction or Error proposal system

References and URLs

[1] Grefenstette Gregory, Tapanainen Pasi, (1994). What is a word, what is a Sentence? Problem of Tokenization.

[2] Cunningham Hamish, Maynard Diana, Bontcheva Kalina, Tablan Valentin, (2002). GATE: An Architecture for Development of Robust HTL Applications.

[3] Zhang Jiasong, El-Gohary M. Nora, (2015). Semantic NLP-based Information Extraction from Construction Regulatory Documents for Automated Compliance Checking.

[4] Bontcheva Kalina, Cunningham Hamish, Tablan Valentin, Maynard Diana, Hamza Oana, (2002). Using GATE as an Environment for Teaching NLP.

[5] Cunningham H., Maynard D., Tablan V., (2000). JAPE: a Java Annotation Patterns Engine.

[6] Hogenboom Alexander, Hogenboom Frederic, Frasincar Flavius, Schouten Kim, Otto van der Meer, (2013). Semantic-based information extraction for detecting economic events.

[7] Brorsje Jethro, Hogenboom Frederik, Frasincar Flavius, (2010). Semi-Automated Financial Events Discovery Based on Lexico-Semantic Patterns.

[8] Akerbergt Hans Svenssont Ola, Schulz Bastian, Nuguest Pierre, (2001). CarSim: An Automatic 3D Text-to-Scene Conversion System Applied to Road Accident Reports.

[9] Sproat Richard, (2001). Inferring the environment in a text-to-scene conversion system.

[10] Minhua Ma, (2006). Automatic Conversion of Natural Language to 3D Animation.

[11] Seversky M. Lee, Yin Lijun, (2006). Realtime Automatic 3D Scene Generation from Natural Language Voice and Text Descriptions.

[12] Coyne Bob, Sproat Richard, Hirschberg Julia, (2010). Spatial Relations in Text-to-Scene Conversion.

[13] Coyne B., Sproat R., (2001). WordsEye: An automatic text-to-scene conversion system.

[14] Adorni Giovanni, Di Manzo Mauro, Giunchiglis Fausto, (1984). Natural Language driven Image Generation.

[15] Arens Michael, Ottlik Artur, Nagel Hans-Hellmut, (2002). Natural Language Texts for a Cognitive Vision System.

[16] IJntema Wouter, Sangers Jordy, Hogenboom Frederik, Frasincar Flavius, (2012). A Lexico-Semantic Pattern Language for Learning Ontology Instances from Text.

[17] Maynard Diana, Bontcheva Kalina, Cunningham Hamish, (2004). Automatic Language-Independent Induction of Gazetteer Lists.

[18] Kibble R., (2013). Introduction to natural language processing.

[19] Staab Steffen, Erdmann Michael, Maedche Alexander, (2000). Semantic Patterns.

[20] Liddy, E. D., (2001). Natural language processing.

[21] Jurafsky Daniel, Martin H. James, (2018). Speech and Language Processing, An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition.

URLS

[22] http://www.web3d.org/wiki/index.php/X3D_and_HTML5

[23] https://gate.ac.uk/books.html

[24] https://www.w3.org/XML/

[25] http://www.web3d.org/x3d

[26] https://www.x3dom.org/

Thanks for your attention !

For more, read the paper:
George A. Krikos, Nikitas N. Karanikolas, George Miaoulis, Athanasios Voulodimos: Greek language object representation scene system: a text interface for designing X3D objects and X3D scenes. PCI 2019: 164-167. Proceedings of the 23rd Pan-Hellenic Conference on Informatics, November 2019, Pages 164–167, https://doi.org/10.1145/3368640.3368666

See also:
Personal Web Page of Professor Nikitas N. Karanikolas
http://users.uniwa.gr/nnk/
http://users.uniwa.gr/nnk/papers/paper_index.htm