PaintTalk: Draw a picture by using natural language

Hunmin Park

December 05, 2019

Contents

Introduction & motivation

Many people enjoy drawing the a picture on the computer. They usually uses mouse or stylus to draw the shapes.

However, we can't use those tools on some devices such as some mobile phones.

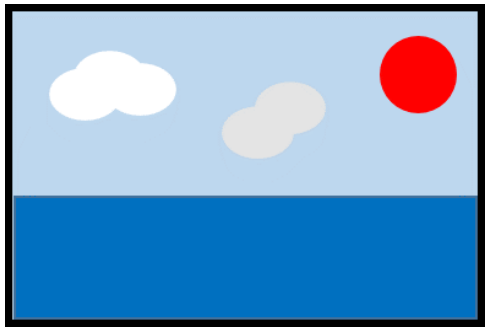
Introduction & motivation

So I propose **PaintTalk**, a 2D graphics editor which uses **natural language** to describe a picture.

Sun is circle and its size is 30.
Color of sun is red.
Position of sun is (180, 30).

Sea is rectangle.
Color of sea is blue.
...

$\Rightarrow$



- By using natural language, we can draw a picture without mouse or stylus.
- Furthermore, compared to the previous methods, natural language input is more understandable and easier to modify.

Big picture

System outline

PaintTalk is a program s.t

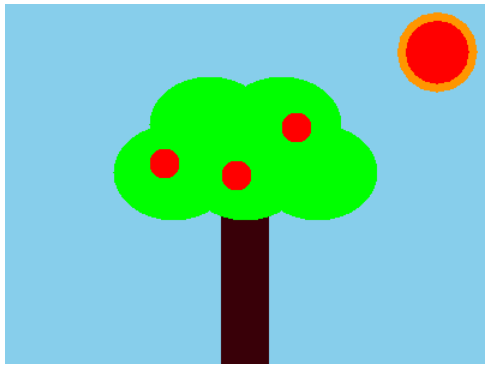
- input is a string
- and output is an image(2D array of RGB values).

Size of canvas is (400, 300)
and its color is (135, 206, 235).

Sun is circle
and its size is 60
and its position is (330, 10)
and its color is red
and size of its border is 7
and color of its border is (255, 150, 0).

...

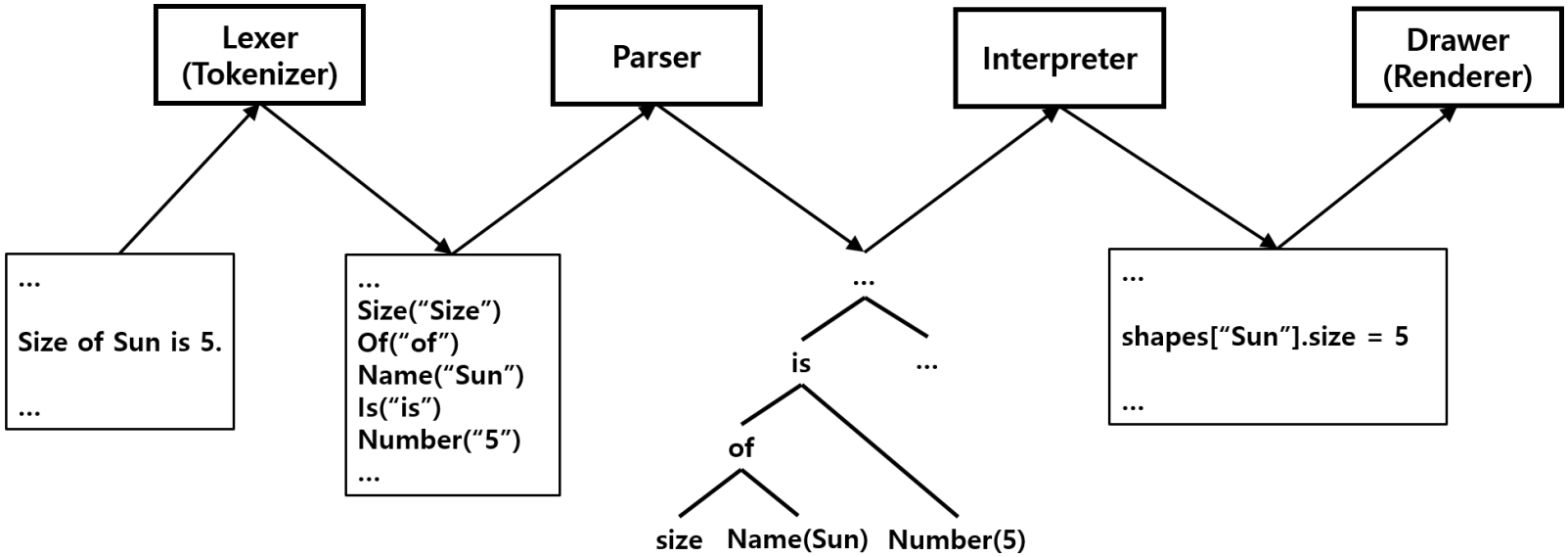
$\Rightarrow$



Big picture

System outline

Structure of the system looks like the following:



Big picture
Grammar

Input consists of the list of **sentences**. The types of sentence are:

Type	**Example**
⟨Name⟩ is ⟨Shape⟩.	Sun **is** circle.
⟨Target⟩ is ⟨Value⟩.	Color **of** border **of** A **is** circle.
⟨Name⟩ is ⟨Order⟩. ⟨Name⟩	A **is in front of** B.

Big picture
Grammar

We can concatenate multiple sentences with **and**.

Sun is circle **and** color of sun is red.

We can avoid repeating the name by using **its**.

Sea is rectangle **and its** color is blue.
Its size is (60, 30) **and** size of **its** border is 5.

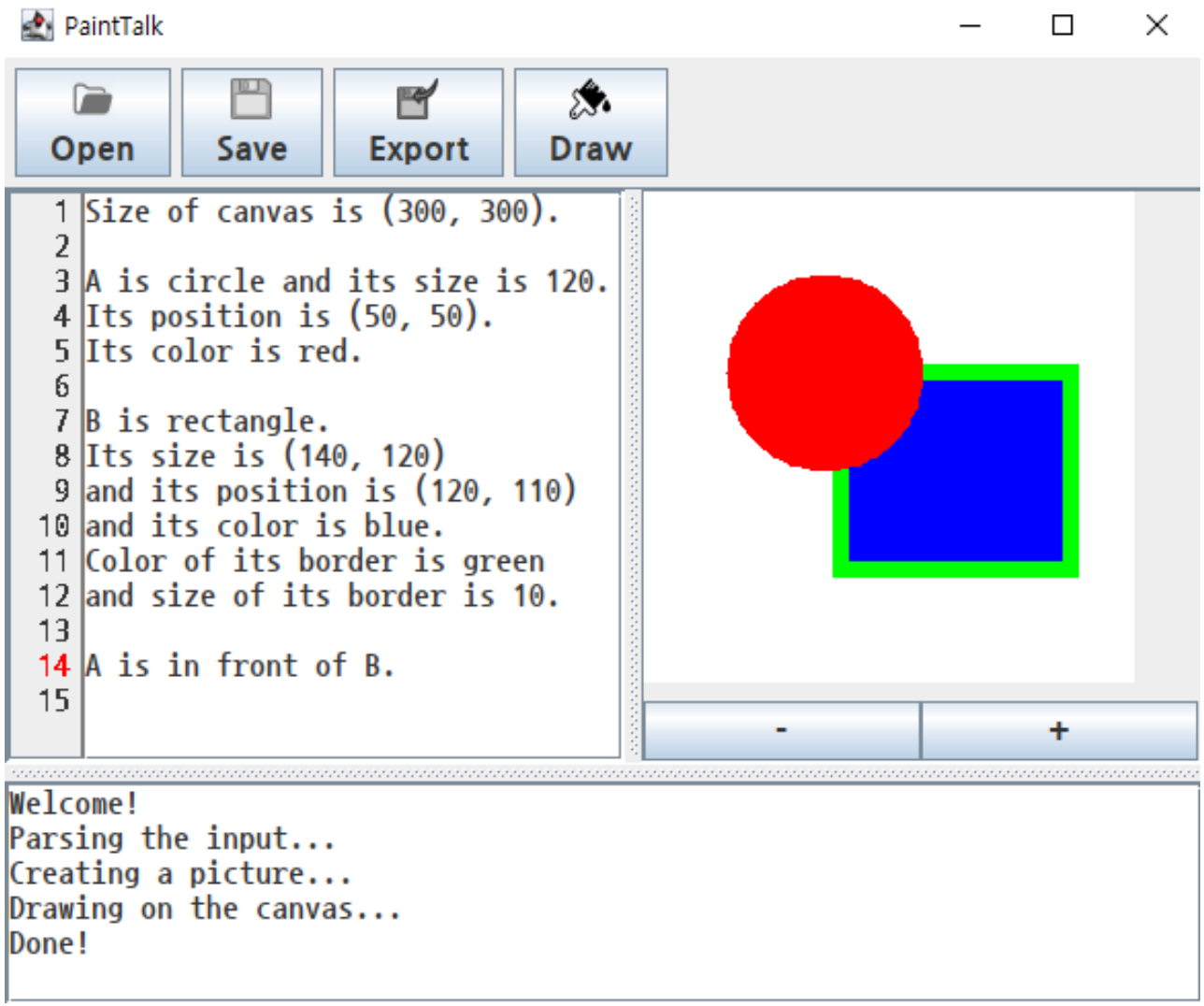
Full grammar. . .

```
<Input> ::= <Sentence> "." (<Sentence> ".")*
<Sentence> ::= <BasicSentence> ("and" <BasicSentence>)*

<BasicSentence> ::= <Name> "is" <Shape>
                  | <Name> "is" <Order> <Name>
                  | <Target> "is" <Value>

<Target> ::= "its" <Attribute>
           | <Attribute> "of" "its" <Area>
           | <Attribute> "of" <Area> "of" <Object>
           | <Attribute> "of" <Object>

<Object> ::= <Name> | <Canvas>
<Value> ::= <Number> | <Tuple> | <Color>
<Canvas> ::= "canvas"
<Name> ::= [a-zA-Z_][a-zA-Z0-9_]*
<Shape> ::= "Circle" | "Square" | ...
<Order> ::= "in" "front" "of" | "behind"
<Attribute> ::= "position" | "size" | ...
<Area> ::= "border"
<Tuple> ::= "(" <Number> ("," <Number>)* ")"
<Number> ::= "3" | "15" | "230" | ...
<Color> ::= "red" | "green" | ...
```

Demo screen shot

Simple shapes

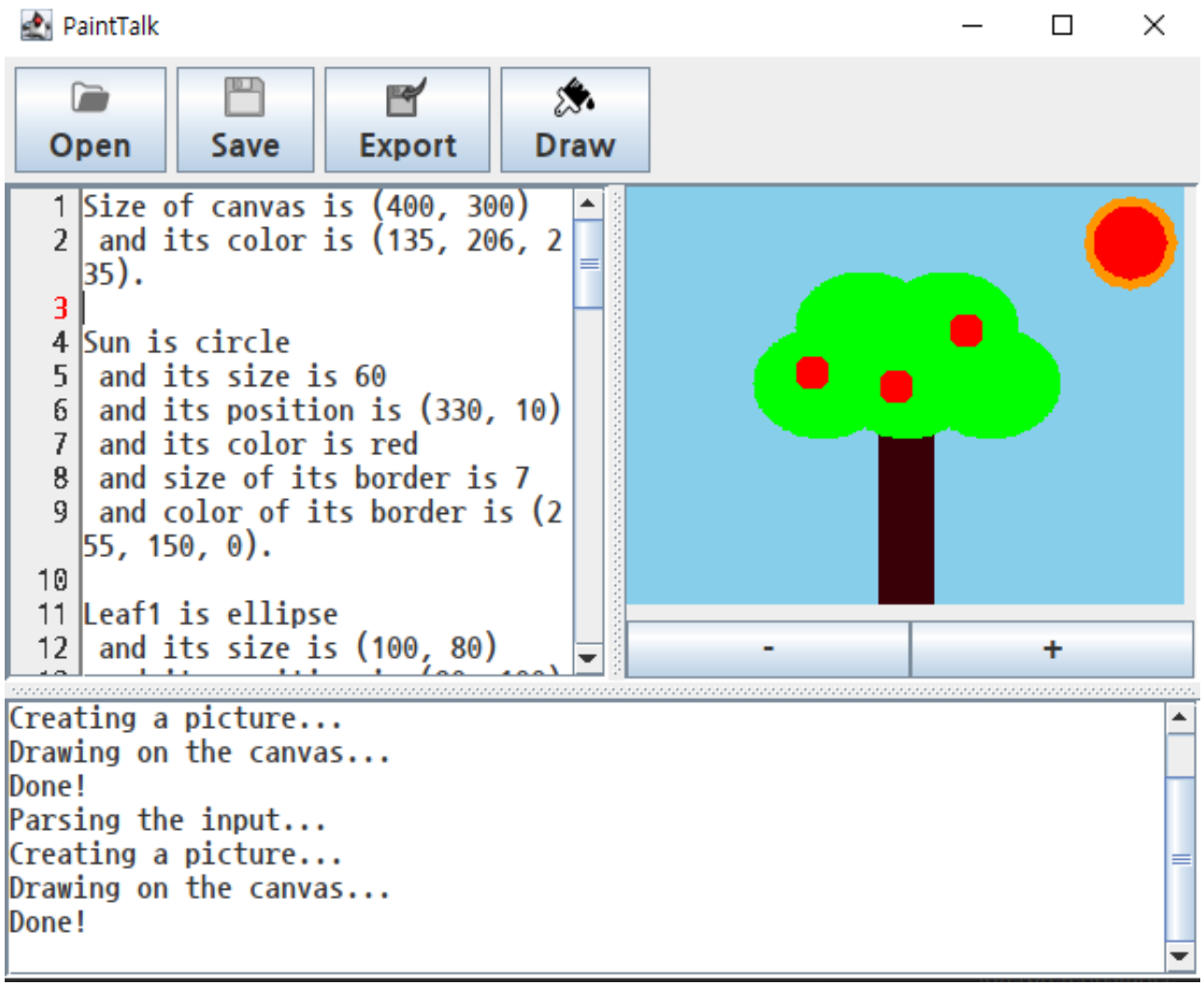
Let's draw some simple shapes.



Demo screen shot

Tree

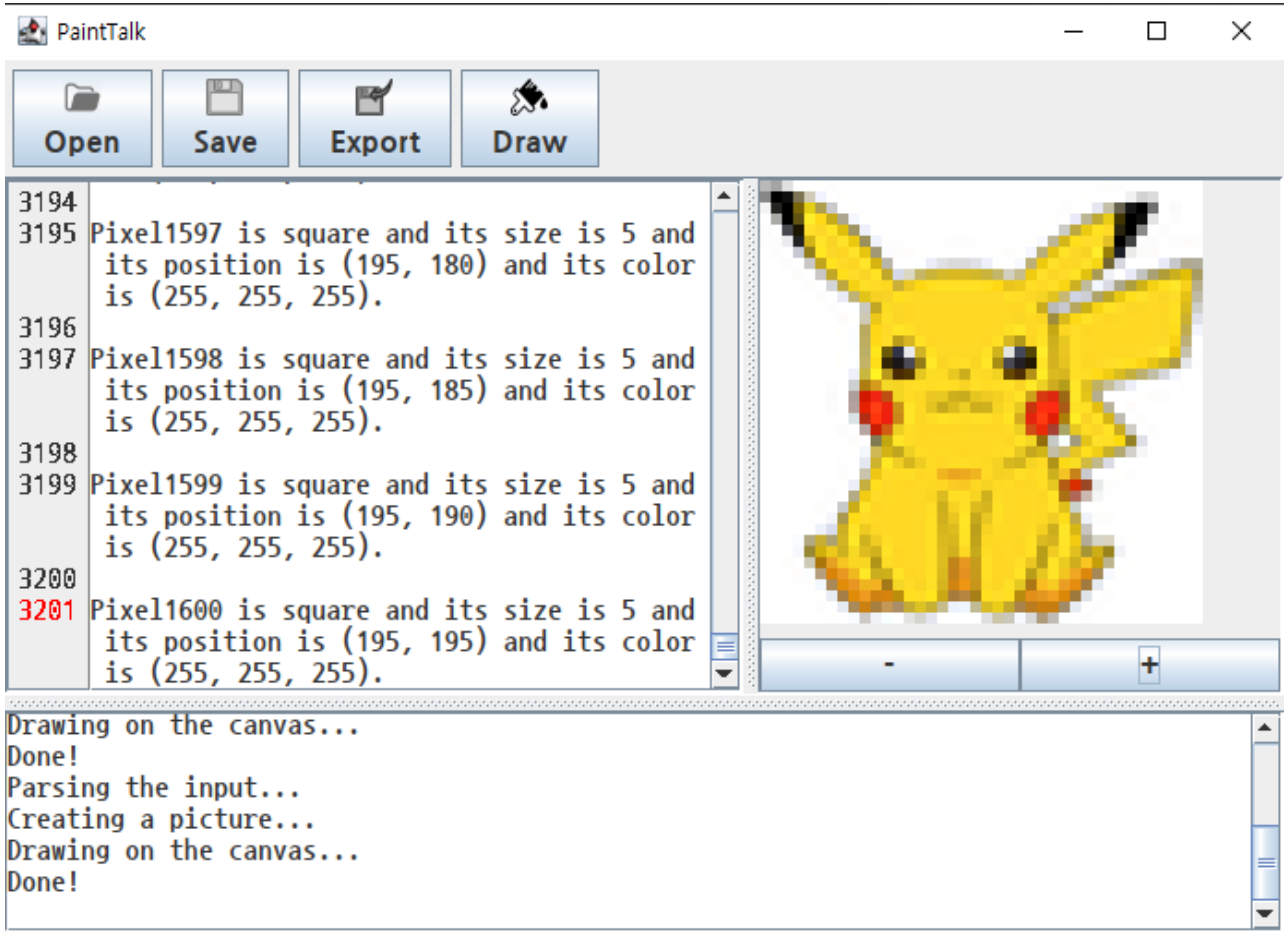
Let's draw more complex one.



Demo screen shot

Pikachu

The program can also handle extremely huge input.

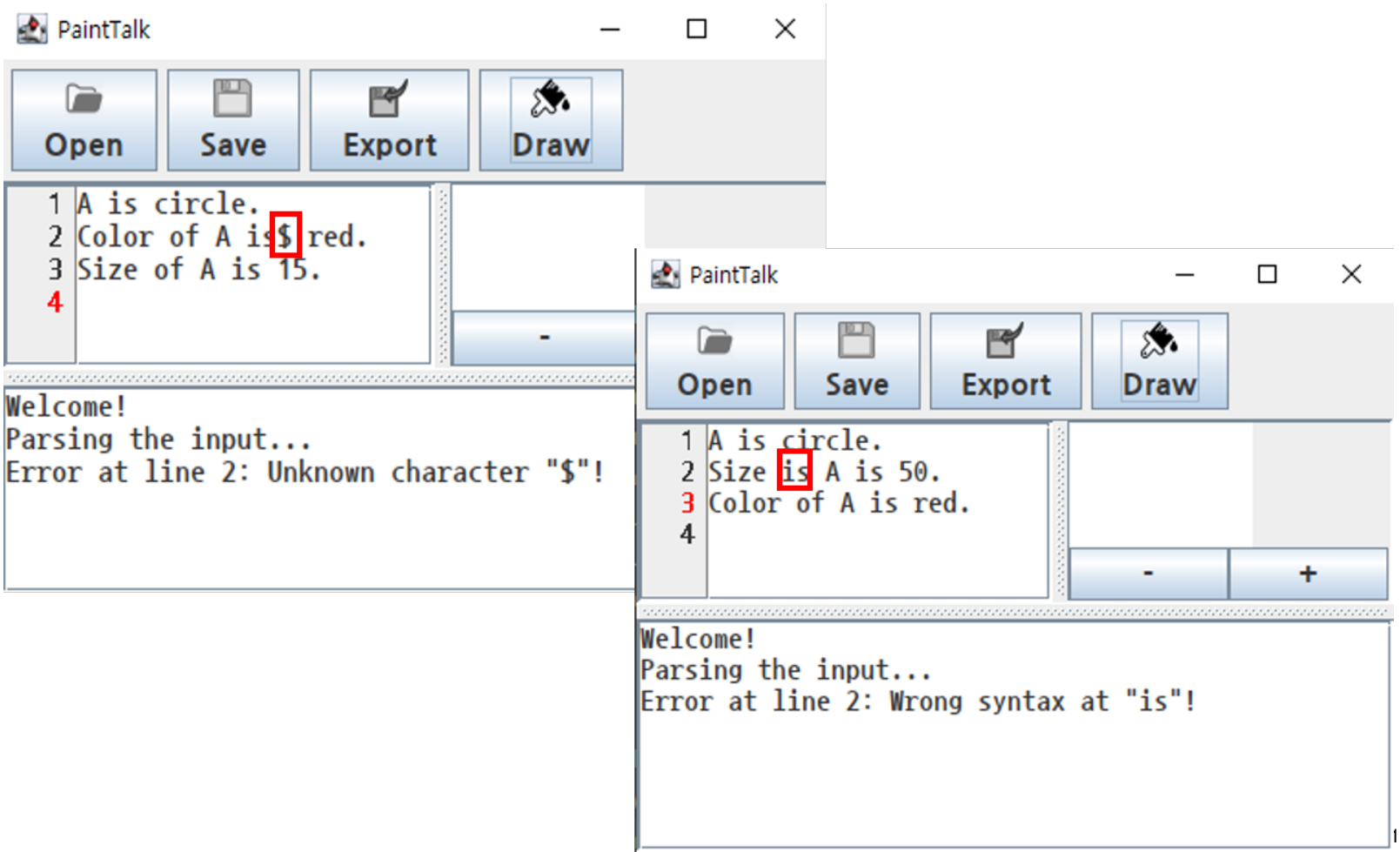


(Of course this input is generated by script, not by hand. . .)

Demo screen shot

Error handling

It also handles the wrong inputs and warns to the user. (ex. Wrong character, wrong syntax, use undefined name, etc.)



Comparison and discussion

Comparison

PaintTalk vs Traditional methods (Mouse or stylus)

Pros of **PaintTalk**

- Don't need mouse or stylus
- Human readable input
- Easier to modify the input
- Store text instead of image, which is much heavier than the text

Cons of **PaintTalk**

- Harder to draw complex images
- Add more features $\rightarrow$ Implementation difficulty $\uparrow$

Comparison and discussion

Discussion

Technical challenges I had

- Construct the appropriate grammar
  (to support various kinds of sentences)
- Error handling in parser and interpreter

Further things we can do

- Support more shapes (ex. Polygon, Curve, etc.)
- Allow less-strict grammar
- . . .

Conclusion

- We can perform 2D painting by using natural language.
- It has several advantages, such as human readability.
- But it also has some weak points.
  ex. Hard to draw complex picture

Conclusion


```
thanks(me, you).
```