

グループ1 発表スライド

Next

B
Color

Blue



Shape

Cube



A
Color

Red



Shape

Cube



C
Color

Green



Shape

Cube



目次

- メンバー紹介
- プランニングの機能拡張について
- 考察
- デモプレイ

メンバー

- 清水涼太 機能拡張
- 梶田大輔 機能拡張
- 島野広大 自然言語処理
- 新海知道 物理演算処理
- 鈴木健太 物理演算処理

システムの概要

- プランニングの拡張
- GUIを3次元空間での表現に変更

システムの仕様

- ブロックの数を3つに制限
- 入力を自然言語で処理
- 実行結果を3Dで表現

システムの特徴

- 色、形を用いたプランニング
- 英語の自然言語に対応
- 3D空間の実現にUnityを使用

実装上の工夫(ブロックの定義)

```
class Item{  
    String name;  
    String color;  
    String shape;  
  
    Item(String theName,  
        String theColor,String  
        theShape){  
        name      = theName;  
        color     = theColor;  
        shape     = theShape;  
    }  
}
```

- Itemクラスの作成
- 目標状態をブロックの名前で書き換える

実装上の工夫(ブロックの定義)

```
public void instatiate(Vector inList, Vector List){  
    for(int i = 0; i < List.size(); ++i){  
        StringTokenizer st = new StringTokenizer((String)List.elementAt(i));  
        String tmp = st.nextToken();  
        String s;  
        if(tmp.equals("ontable")){  
            s = st.nextToken();  
            inList.add(i, "ontable " + search(s));  
            System.out.println(inList.get(i));  
        }else if(tmp.equals("clear")){  
            s = st.nextToken();  
            inList.add(i, "clear " + search(s));  
            System.out.println(inList.get(i));  
        }else if(tmp.equals("handEmpty")){  
            inList.add(i, "handEmpty");  
            System.out.println(inList.get(i));  
        }else{  
            String tmp2 = search(tmp);  
            s = st.nextToken(); //on  
            s = st.nextToken();  
            inList.add(i, tmp2 + " on " + search(s));  
            System.out.println(inList.get(i));  
        }  
    }  
}
```

- Itemクラスの作成
- 目標状態をブロックの名前で書き換える

実装上の工夫(ブロックの定義)

```
public String search(String s){  
    String answer = null;  
  
    for(int i=0;i < items.size();+  
        +i){  
  
        Items I =  
            ((Items)items.elementAt(i));  
  
        if(I.name.equals(s) ||  
           I.color.equals(s) ||  
           I.shape.equals(s)){  
  
            answer = I.name;  
  
        }  
  
    }  
  
    return answer;  
}
```

- Itemクラスの作成
- 目標状態をブロックの名前で書き換える

実装上の工夫(ブロックの定義)

- 目標状態(変更前)

red on blue

ontable cube

clear triangle

- 目標状態(変更後)

B on A

ontable A

clear B

名前	色	形
A	blue	cube
B	red	triangle

考察(ブロックの定義)

- Ontable cube
- red on blue
- blue on red
- clear triangle
- handEmpty
- Ontable A
- C on A
- B on C
- clear B
- handEmpty

名前	色	形
A	blue	cube
B	blue	triangle
C	red	trapezoid

考察(ブロックの定義)

- Ontable cube
- red on blue
- blue on red
- clear triangle
- handEmpty
- Ontable A
- C on **B**
- B on C
- clear B
- handEmpty

名前	色	形
A	blue	cube
B	blue	triangle
C	red	trapezoid

考察(ブロックの定義)

- 今のアルゴリズムでは同じ色、形に対して同じ名前になってしまう。

もし正しい結果にするためには 2^n 通り考えなくてはいけないので実装を諦めた。

- 三角形ブロックの制約については、目標状態になる場合にしか「Plase ?X on ?Y」を実行しないため、考えなくて良い。

実装上の工夫(自然言語処理)

- ブロックの**状態**を示す自然言語の処理を行うメソッド
- ブロックの**操作**を示す自然言語の処理を行うメソッド

プログラムの記述を簡潔にし
自然言語の正確な処理の成功率を上げる

実装上の工夫(自然言語処理)

- ブロックの名前、ブロック操作に用いられる動詞などのキーワードを格納した配列
- キーワードの有無や文に含まれるブロックの数によって判別

多様な表現を許容することが可能

実装上の工夫(自然

例： Would you put A on B ?

Would you **put** A on B ?

Would you put **A** on B ?

Would you put A on **B** ?

→ **Place A on B** と判別

Place ?x on ?y

Place ?x on ?y

Place **A** on ?y

Place **A** on **B**

put A down on the table
remove ?x from top on ?y
pick up ?x from the table

考察(自然言語処理)

- 二つのメソッドに分割した
 - 記述は簡潔になるが、プランニングに組み込む作業では構造を考慮する必要がある
- キーワードを利用して多様な表現を許容
 - キーワードの追加は容易であるが、プログラマ側の意図していないキーワードには対応できない
 - ユーザー側から利用するキーワードを学習する機能

3D空間での物理演算 システムの仕様

- 前述の機能をUnity上で実装した

-point-

- Java言語をC#に書き換える
- Blockの形,色をGUI上で変更する
- 初期状態、目標状態をGUI上で変更する
- PlanningをGUI上で実行する

3D空間での物理演算 実装上の工夫

C#に存在しない型、処理が異なる型を
別の型に書き換え正常に処理が行われ
るように変更した

変更したもの

- Planning
- 自然言語処理 メソッド

3D空間での物理演算 実装上の工夫

見やすいようにUIを整えた

デモを見てください

3D空間での物理演算 考察

- Unityのプロジェクト、シーンを複数人で
マージする上でいくつか不具合が生じた
- 3D空間を利用したが物理演算を活用
した利点を生かし切れなかった
(物理演算を切っても問題ない設計)

3D空間での物理演算 考察

- Javaで出来ていることがC#で実現しきれずに終わってしまった。
- 三角形の形と四角形に対応して、他の形には対応しきれないものとなった。
- どのブロックがA,B,Cか非常に分かりにくいものとなってしまった。

感想

親しい友人とのグループ課題でしたが、情報伝達が上手くいかずスムーズに課題が進められなかったので、グループワークの難しさを感じました。

この経験を次の機会に生かしていけると良いと思います。

ご試聴ありがとうございました。