



# Roboko: ソースコードに 写真を貼り込める 統合開発環境

加藤 淳, 坂本 大介,  
五十嵐 健夫

東京大学, JSPS

```
void draw() {  
    if (robot.isActing()) {  
        if (flag) {  
  
            robot.setPose();  
        } else {  
  
            robot.setPose();  
        }  
    }  
    flag = !flag;  
}
```

# はじめに

## 姿勢情報を用いたプログラミングは難しい



踊り



餅つき



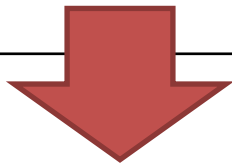
姿勢認識

Images retrieved from <http://www.youtube.com/watch?v=Z1TK4D1Jr6k>  
and <http://www.robospot.jp/cgi-bin/diarypro/diary.cgi?no=94>

# 既存手法

## 文字ベースのプログラミング

```
joint[0].setAngle(0.6);      arm.setPosition(4.3, 2.4);  
joint[1].setAngle(0.7);  
joint[2].setAngle(-0.24);   robot1.setPose(pose32);  
...
```



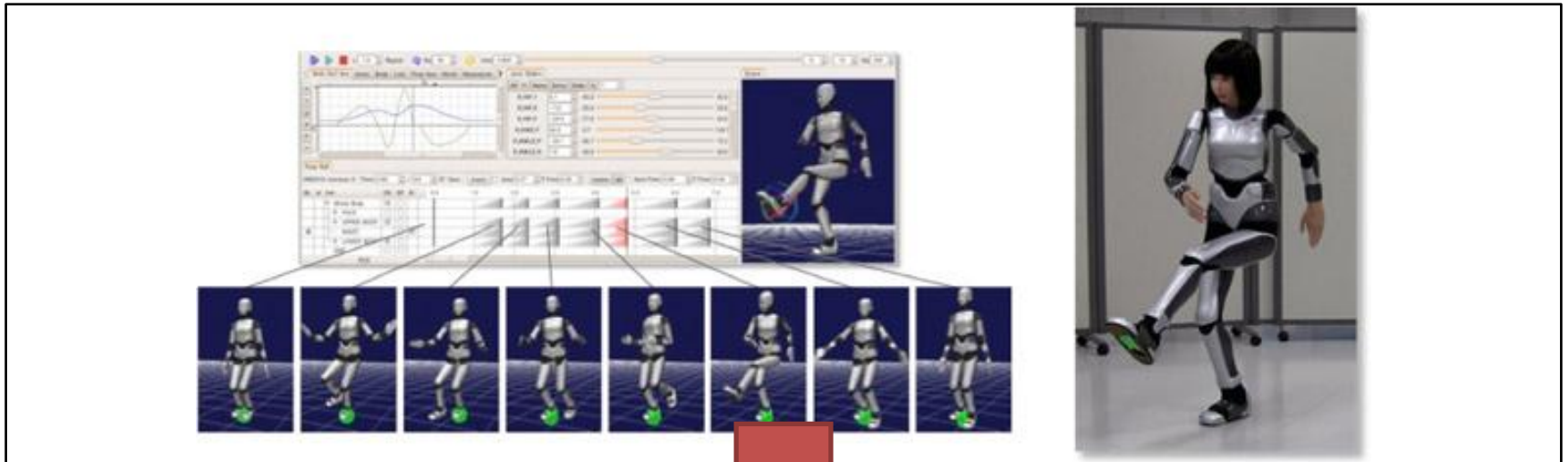
### 問題:

実際の姿勢がどうなっているのか  
想像しにくい



# 既存手法

## GUIベースのエディタ



Choreonoid [中岡 et al., 2011]

**問題:**

ユーザとインタラクションできない  
制御構造を書けない

# 提案手法

**Roboko:**

ソースコードに  
写真を貼り込める  
統合開発環境

```
void draw() {  
    if (robot.isActing()) {  
        if (flag) {
```



```
        robot.setPose(  
    } else {
```

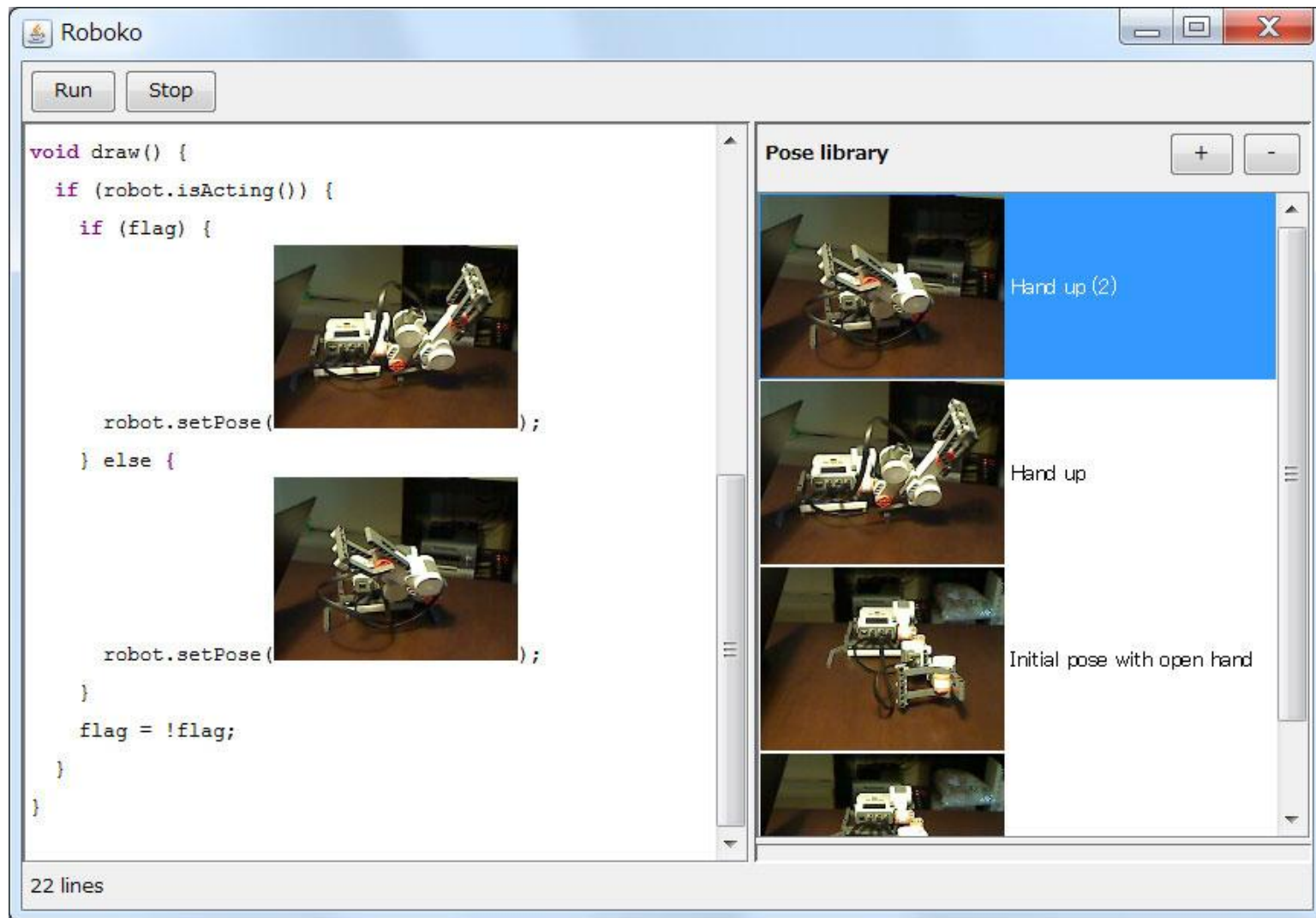


```
        robot.setPose(  
    }
```

```
}  
flag = !flag;
```

```
}
```

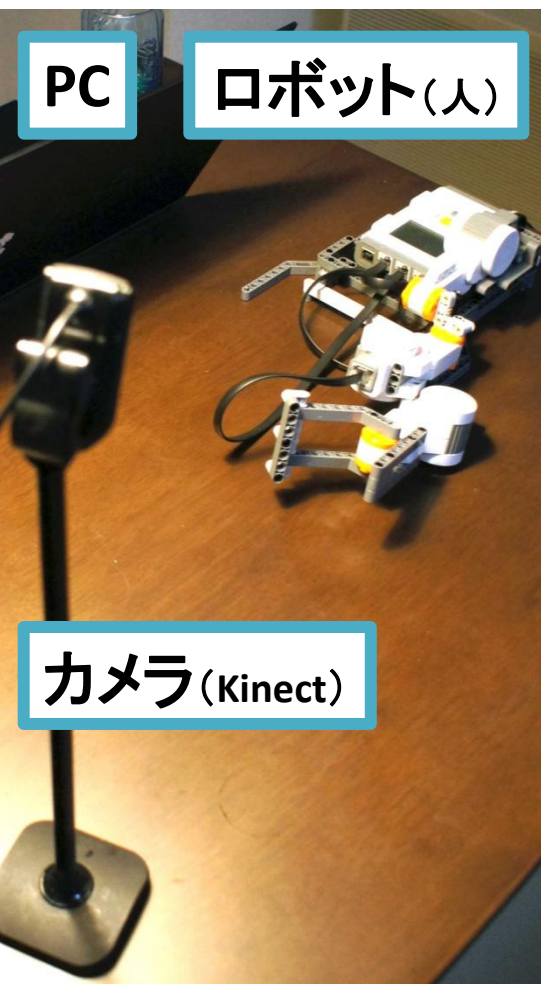
# 開発環境の概観





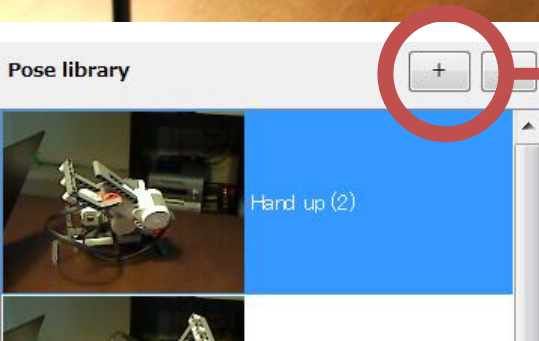
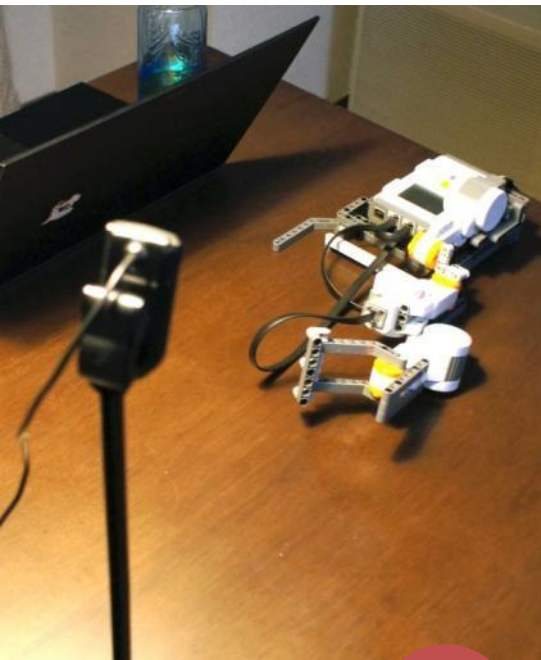
# 姿勢情報のキャプチャ (1/3)

## ハードウェアセットアップ



# 姿勢情報のキャプチャ (2/3)

## “+”ボタンをクリックする

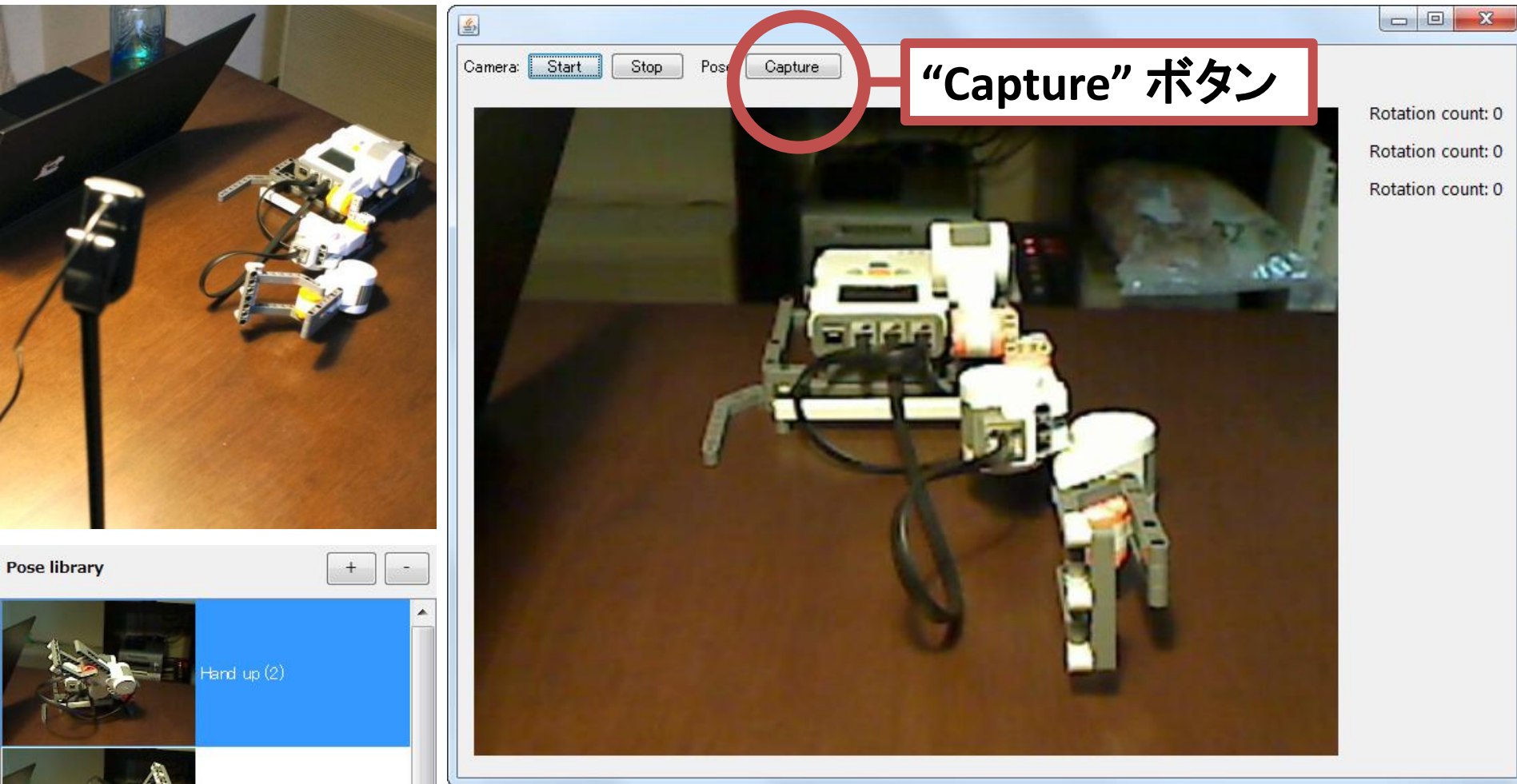


“+” ボタン



# 姿勢情報のキャプチャ (3/3)

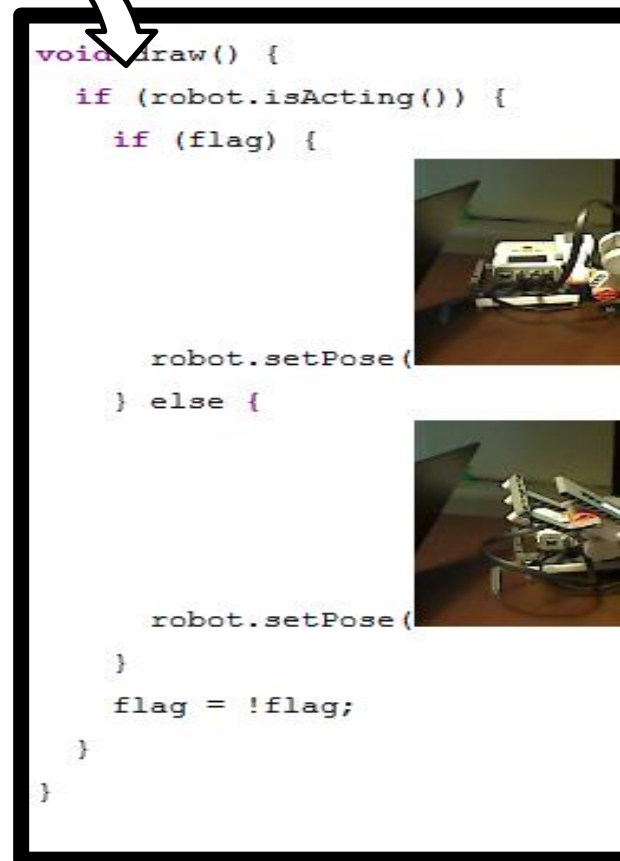
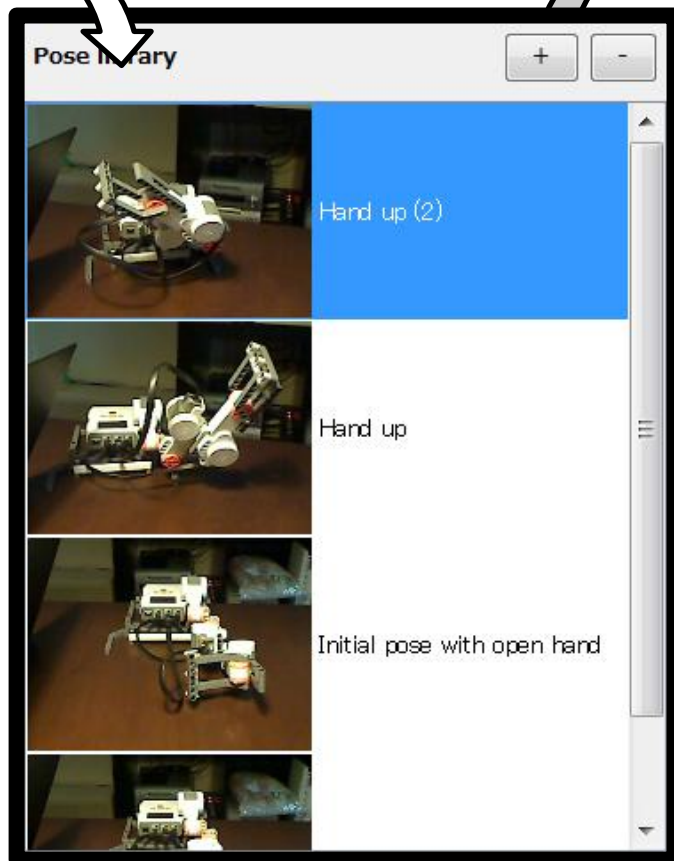
## “Capture”ボタンをクリックする



# プログラマのワークフロー

## 1. キャプチャ

## 2. ドラッグ&ドロップ



# 写真を引数に取れるAPI (1/2)

## 姿勢情報の取得と設定



```
robot.getPose().eq(, 0.04)
```



```
robot.setPose( );
```

# 写真を引数に取れるAPI (2/2)

## 連続した姿勢の制御

```
Action a = robot.action();
```



```
a = a.pose(  
a.play();  
) .wait() .pose(  
a.play();
```



# 関連研究 (1/2)

## Barista [Ko et al., 2006]

### 絵などを貼り込める構造化エディタ

```
public class ImageTransformer  
{
```

Rotates an **Image** by an angle given in degrees.

For example,

```
transformer.rotate(img, 45)
```

performs this operation:

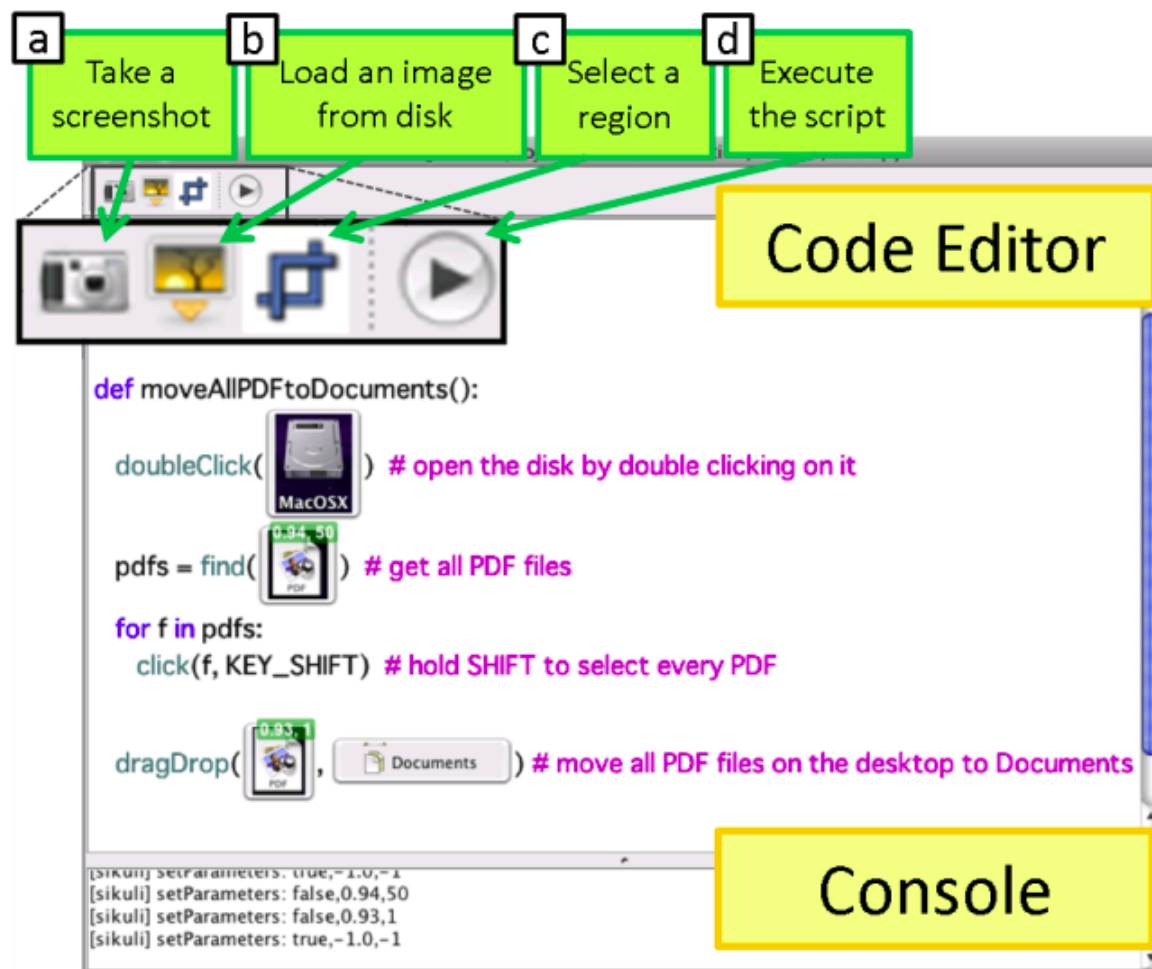


```
public void rotate(Image img, double degrees)  
{
```

# 関連研究 (2/2)

**Sikuli** [Yeh et al., 2009]

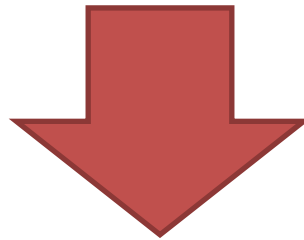
スクリーンキャプチャを貼る開発環境





# 写真を使って何が嬉しいのか？

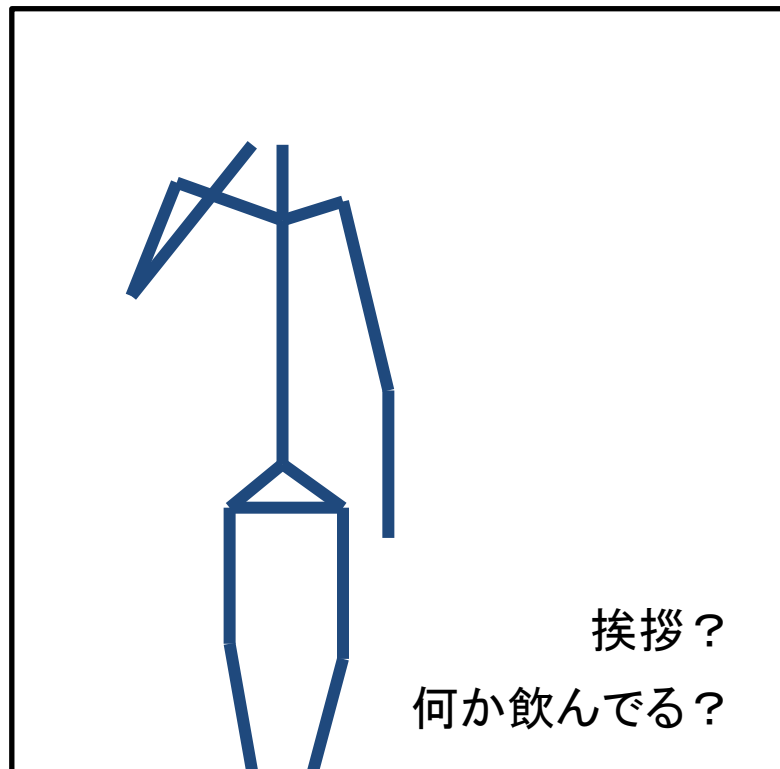
- 3Dモデルとして可視化するほうがストレートだし、それでいいのでは？
- 冗長な情報に何の意味があるのか？



1. 周辺環境情報が役立つことがある
2. ロボットの3Dモデルを作る手間が省ける
3. 写真を撮るのは楽しい(誰にでもできる)

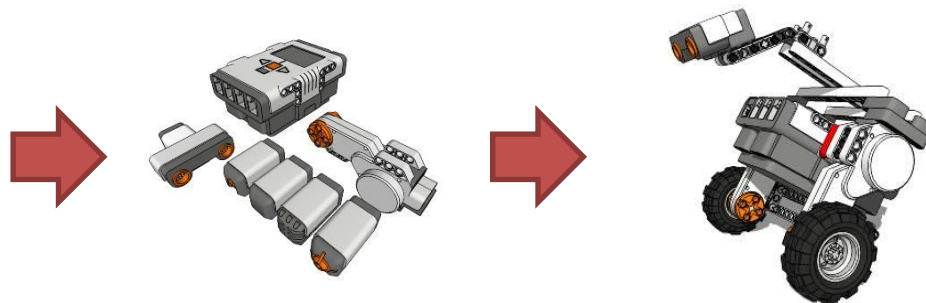
# 写真を使って何が嬉しいのか？ (1/3)

## 周辺の環境情報が役立つことがある



# 写真を使って何が嬉しいのか？ (2/3)

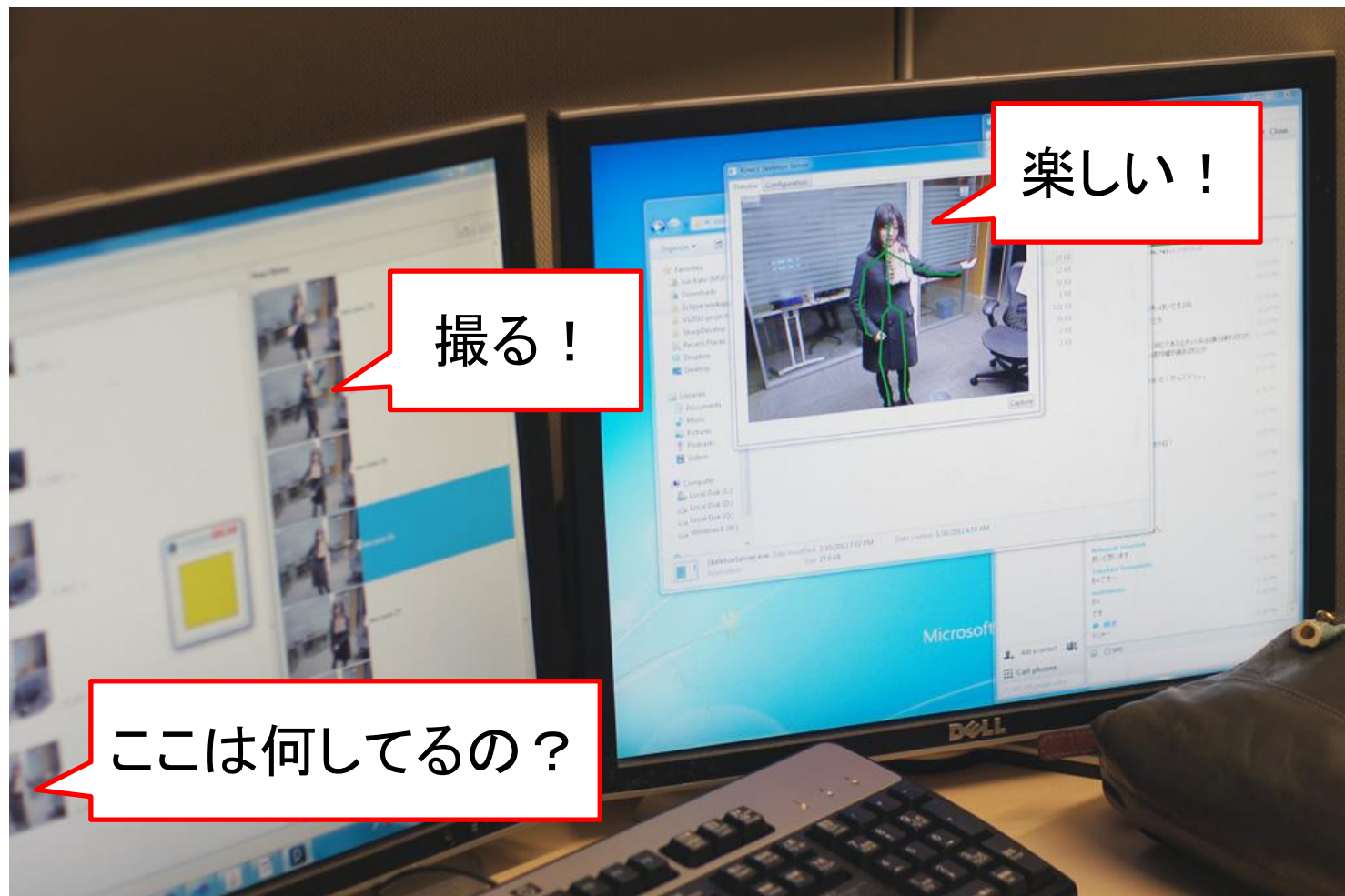
## ロボットの3Dモデルを作る手間が省ける



Robot model retrieved from <http://sketchup.google.com/3dwarehouse/details?mid=c0c6e16583fb387eaa024306630494b>

# 写真を使って何が嬉しいのか？ (3/3)

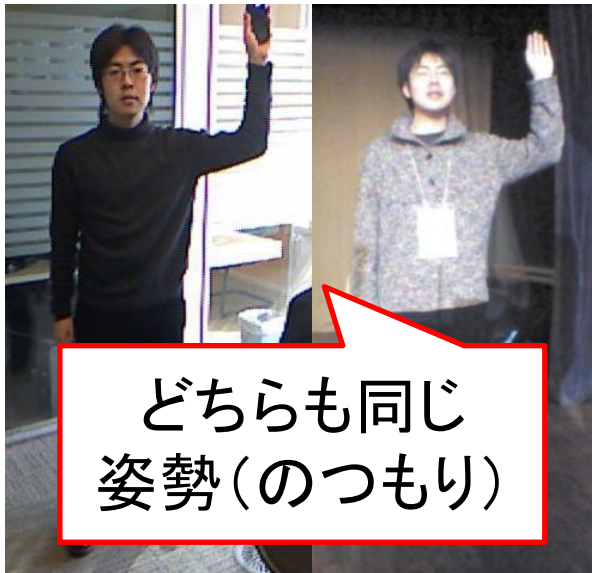
## 写真を撮るのは楽しい(誰にでもできる)





# 今後の課題



- 機械学習による柔軟な姿勢判断
- 姿勢の部分比較
- 動画対応
- ハードウェアのプロトタイピング用途への転用



# Roboko: ソースコードに 写真を貼り込める 統合開発環境

加藤 淳, 坂本 大介,  
五十嵐 健夫

東京大学, JSPS

```
void draw() {  
    if (robot.isActing()) {  
        if (flag) {  
  
            robot.setPose();  
        } else {  
  
            robot.setPose();  
        }  
    }  
    flag = !flag;  
}
```