Visionsketch: Gesture-based Language for End-user Computer Vision Programming

JUN KATO
IGARASHI LAB., THE UNIVERSITY OF TOKYO
http://junkato.jp/visionsketch/
WHAT IS VISIONSKETCH?

Visionsketch language allows end-users (= people without knowledge of programming)

- to extract useful information from images/videos
- to make programs that can detect interesting events from live camera input

by

- building image processing pipelines
- with drawing shapes and choosing primitives
- without typing text
WHAT IS VISIONSKETCH?

Visionsketch language allows end-users (= people without knowledge of programming)

- to extract useful information from images/videos
- to make programs that are monitoring events from live camera
  by
- building in image processing pipelines
- with drawing shapes and choosing primitives
- without typing text

Let’s go visual!

a quick demo follows
VISIONSKETCH LANGUAGE PRIMITIVES
DESIGNED ACCORDING TO USER INTERVIEWS

**Geometric transform**
- **Linear-polar conversion**
- **Perspective warp**

- *in: any image, out: image of same type*

**Information filtering**
- *in: any image, out: bin image*

**Timelapse conversion**

- *in: any image, out: image of same type*

**Contour counting**
- *in: bin image, out: image + numbers*
VISIONSKETCH LANGUAGE PRIMITIVES
DESIGNED ACCORDING TO USER INTERVIEWS

**Geometric transform**

1) **Deform** region of interest (ROI) to make further processing easier

*in*: any image, *out*: image of same type

**Information filtering**

2) **Reduce** amount of info in ROI

*in*: any image, *out*: bin image

**Timelapse conversion**

3) **Project time** into two-dimensional space ("for" loop)

*in*: any image, *out*: image of same type

**Contour counting**

4) **Extract metadata** hidden behind the concrete image

*in*: bin image, *out*: image + numbers
SHARING SOME CONCEPTS
WITH TEXT-BASED IDE

Comments in code
Text comment →
Freehand annotation

Code completion
Type-based completion →
Parameter-based completion
PROGRAMMING LANGUAGE FOR “PEN & TOUCH” ERA?

Text is a good way to write program with a keyboard. With pen & touch, we can’t input text as before.

While TouchDevelop does good work with its software keyboard… 😊
RELATED WORK (1)
VISUAL REPRESENTATIONS IN IDE

Concrete data integrated in programming environment

**Active Code Completion**
[Omar et al., ACM/IEEE ICSE ‘12]

```
public Color getDefaultColor() {
    return navy;
}
```

**DejaVu IDE**
[Kato et al., ACM UIST ‘12]

**Code completion enhancement**

**Picode IDE**
[Kato et al., ACM CHI ‘13]

```
Pose pose = human.getPose();

if (pose.eq(θ)) { /* do sth */ }
```

**Code editor enhancement**

**Debugger enhancement**
RELATED WORK (2)

LIVE PROGRAMMING

Direct manipulation of program

**TouchDevelop** for GUI

**Excel** for spreadsheet calculation

**PureData** for audio processing

**Word?** for HTML+CSS editing?
FROM USER INTERFACE TO PROGRAMMING LANGUAGE

There is smooth gradation rather than deep valley.

**User interface** = programming language?  **No.** There’s no abstraction.

**HTML** = programming language?  **Probably… no?**
It’s not turing-complete.
(While HTML + CSS3 are! 😊)

**Visionsketch** = programming language?

**SQL** = programming language?  **Yes,** while the domain is limited.

**C, C++, Java, ...** = programming language?  **Definitely yes!**
MELTING THE BOUNDARY BETWEEN UI AND PL

My research contributions:

• Live programming of image processing programs with visual representations

• Bringing UI perspective to PL (User-centered design of touch-optimized language)

• Exporting PL techniques to UI world (Language primitives, IDE, code completion…)

User interface and programming language are both computational languages and can share many stuff 😊
VISIONSKETCH:
GESTURE-BASED LANGUAGE FOR
END-USER COMPUTER VISION PROGRAMMING

APPENDIX
Symbolic representations of program code

These do not fully benefit from pen & touch... code elements are still something symbolic.

We should be able to **draw** something concrete.
FROM USER INTERFACE TO PROGRAMMING LANGUAGE

Every one of these is \textit{language} = \{syntax + words\}

- to make the computer work for us
- designed to balance easiness and degree-of-freedom

\textbf{User interface} = language

\textbf{HTML} = language

\textbf{SQL} = language

\textbf{C, C++, Java, ...} = language

GUI components + possible operations

HTML spec + HTML tags

Syntax + statements

Syntax + statements