Integrated Visual Representations for Programming with Real-world Input and Output

Jun Kato <i@junkato.jp>
Supervised by
Prof. **Takeo Igarashi**,
The University of Tokyo

Background

Programs with real-world input and output (I/O) have become popular.

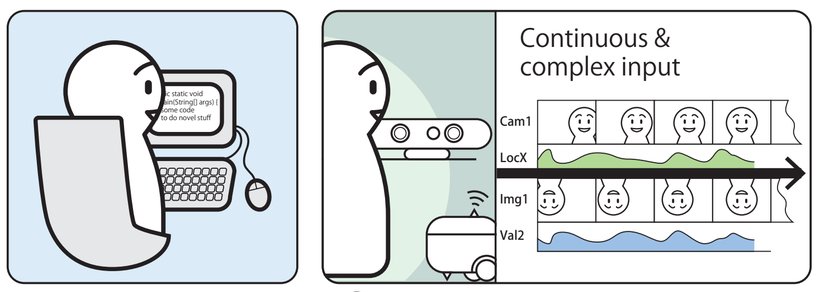
Interactive image processing | Posture data processing | Mobile robot control | Robot posture control

Problem

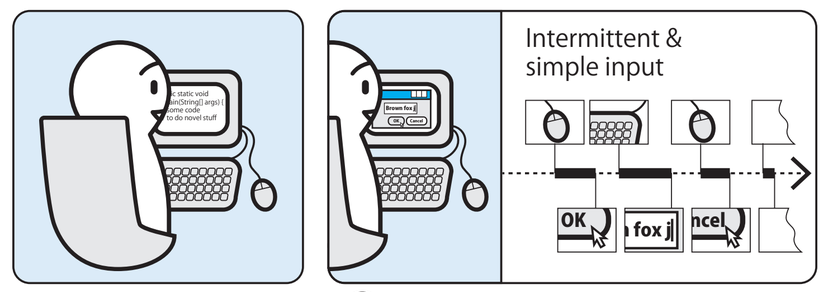
Gulf of execution and evaluation in programming has been deepened.

⬇

Programming with example data is needed.



Development of the programs with real-world I/O



Development of the programs with conventional I/O

Our approach: *integrate graphical representations of the real world into text-based IDEs*

Existing approaches: *Programming by Example* is good for the end-user but not always the best for the programmer.
Text-based programming + external tools help the programmer but could be better when integrated.

Related work: *Visual Programming* focuses on new kinds of visualization of program code and data while we investigate use of existing graphical representations: photos and videos.

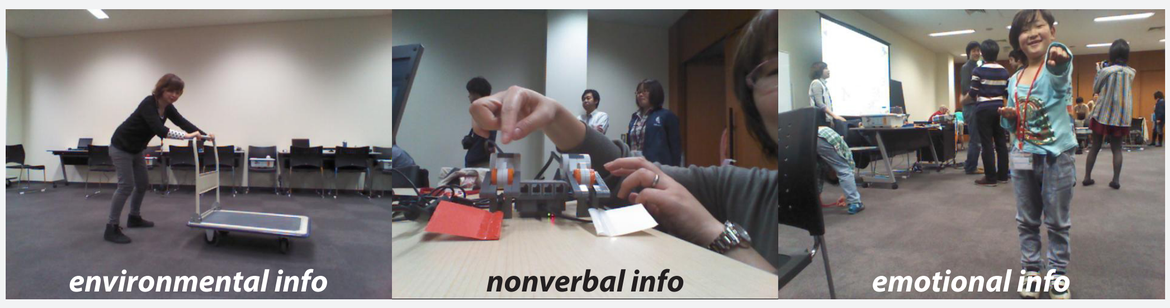
So, how can we integrate graphical representations?
We model the program as follows and assign graphical representations to each component.

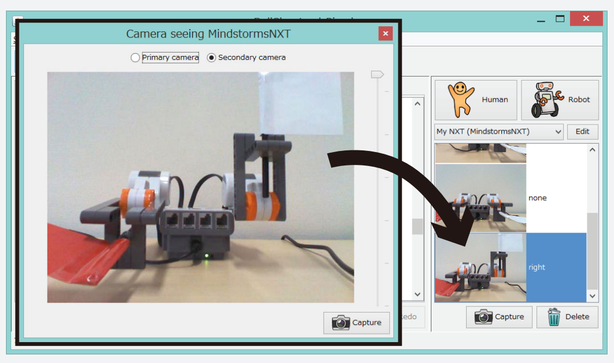
$$out = f(in, c)$$ — where

- ***c*: constants** - static input to the program
- ***in, out*: variables** - dynamic input and output of the program
- ***f*: functions** - specification of the program

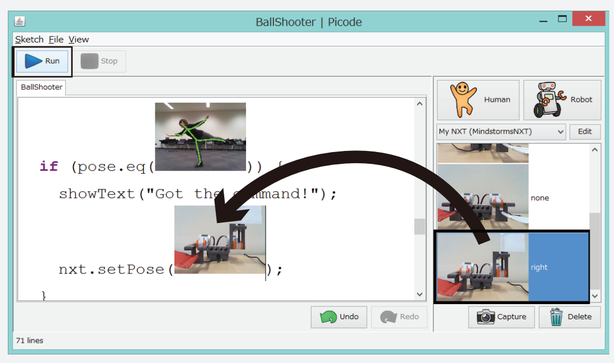
Photos for understanding static data used in the program

Graphical representations of constants, [Kato et al., CHI'13]





(1) capture **example** postures

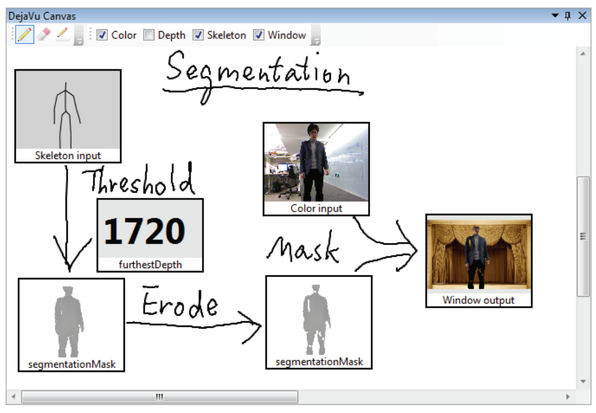


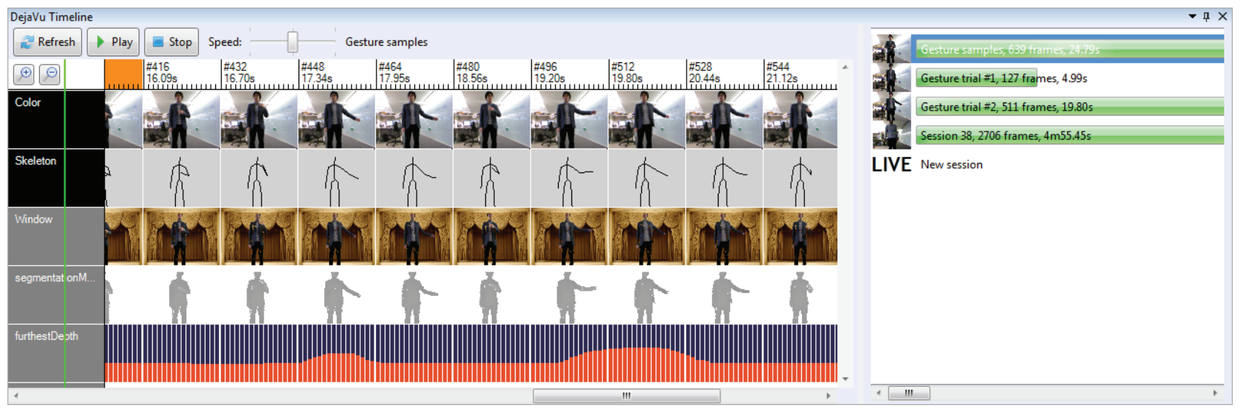
(2) write code with the examples

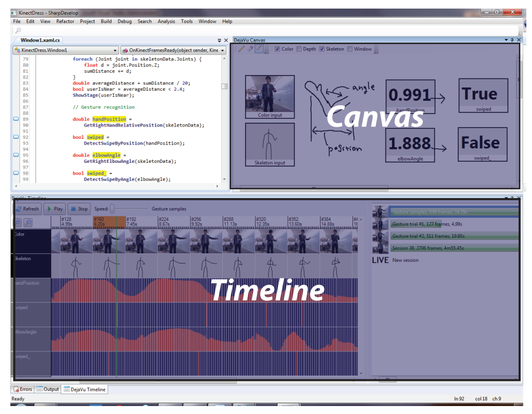
Videos for understanding dynamic behavior of the program

Graphical representations of variables, [Kato et al., UIST'12]

(1) record **example** program execution (2) replay to review the execution
(3) rewrite code to update implementation
(4) update results by re-executing the program with recorded data



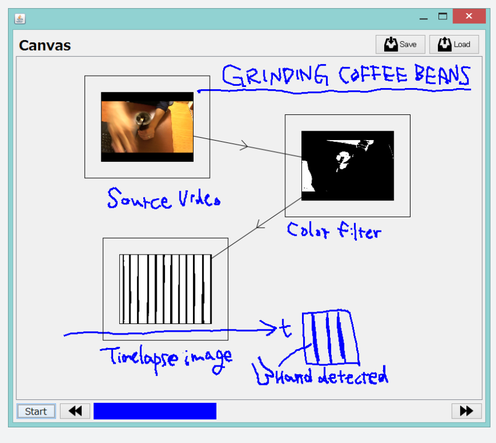


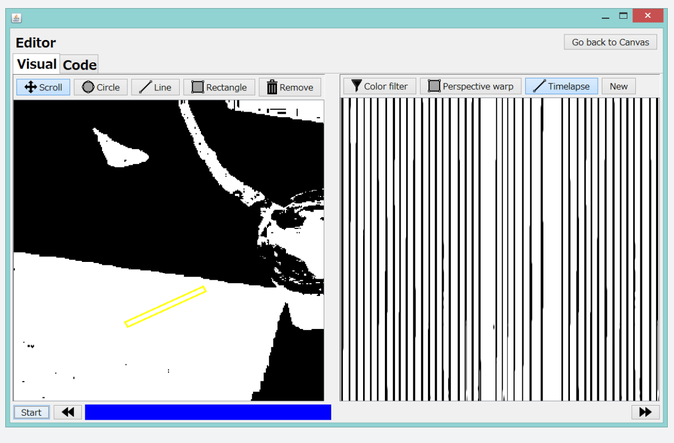


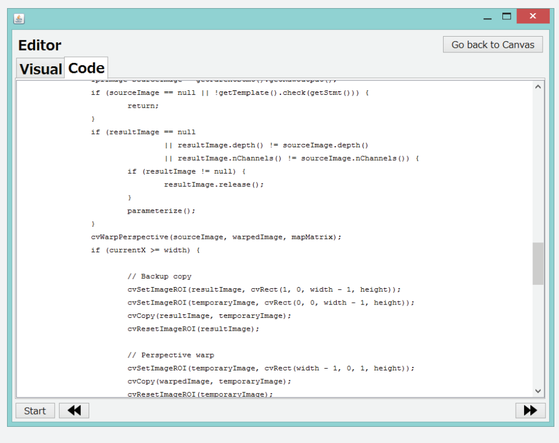
Annotations on photos & videos for specifying program behavior

Graphical representations of functions

(1) create an image processing component by annotating **example** video input
(2) connect components to build a graph and see it in action by playing the video
(3) edit component implementation if needed







Future outlook

3D graphical representations, Multimodal programming, Everyone as a programmer, Liveness in programming

References

Jun Kato, Sean McDirmid, Xiang Cao, "DejaVu: Integrated Support for Developing Interactive Camera-Based Programs", In *Proc. UIST '12*, pp.189-196.
Jun Kato, Daisuke Sakamoto, Takeo Igarashi, "Picode: Inline Photos Representing Posture Data in Source Code", In *Proc. CHI '13*, pp.3097-3100.