Enabling Independent Learning of Programming Concepts through Programming Completion Puzzles

Kyle J. Harms, Noah Rowlett, Caitlin Kelleher

Washington University in St. Louis

Novices Learning Programming - Classroom

Secondary schools offering introductory (or pre-AP) Computer Science courses, change from 2005 baseline

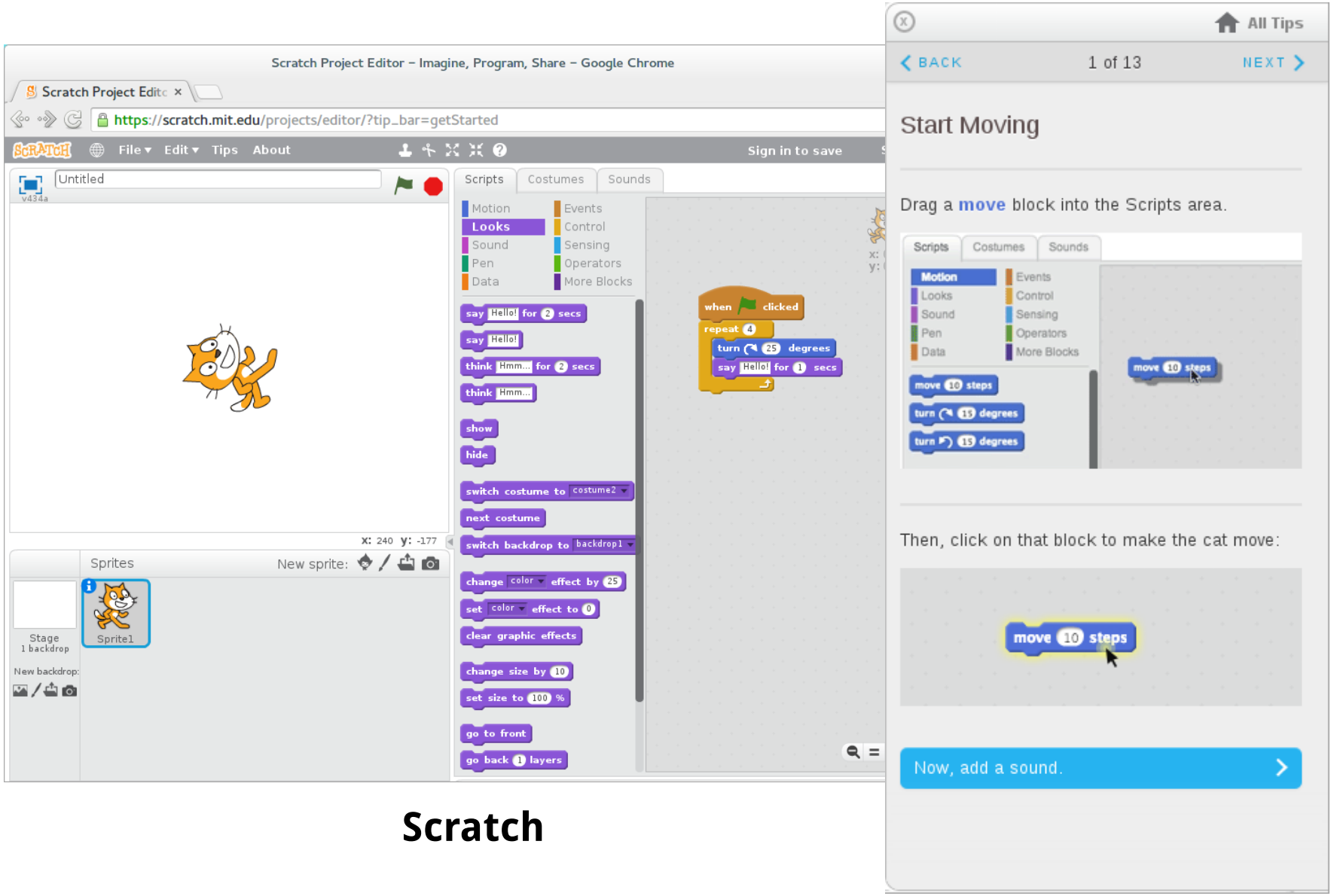
	2007	2009
Yes	-6%	-17%

Secondary offering AP Computer Science courses, change from 2005 baseline

	2007	2009
Yes	-20%	-35%

ACM Computer Science Teachers Association. “Running On Empty: The Failure to Teach K-12 Computer Science in the Digital Age.” http://runningonempty.acm.org/
Microsoft. “A National Talent Strategy” http://www.microsoft.com/en-us/news/download/presskits/citizenship/MSNTS.pdf

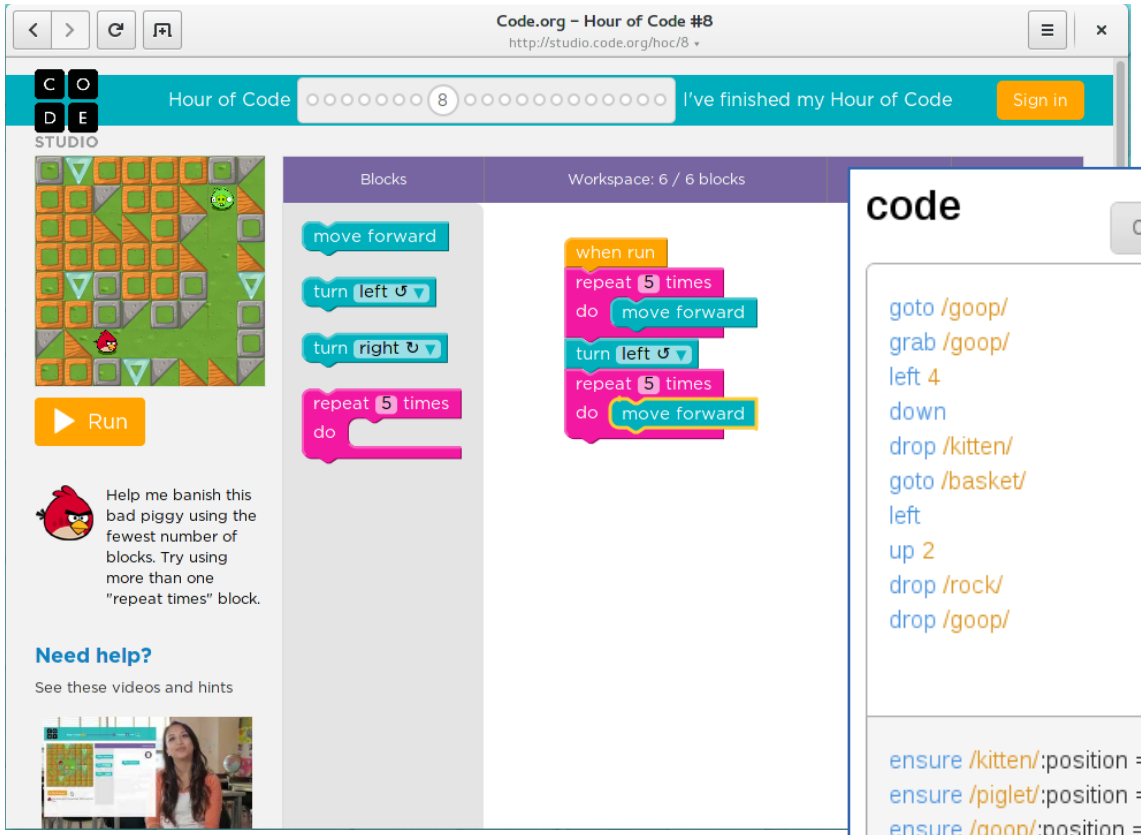
Learning Programming Independently - Tutorials



Scratch

"Scratch," Scratch. [Online]. Available: https://scratch.mit.edu/.

Learning Programming Independently - Puzzle-Like Systems

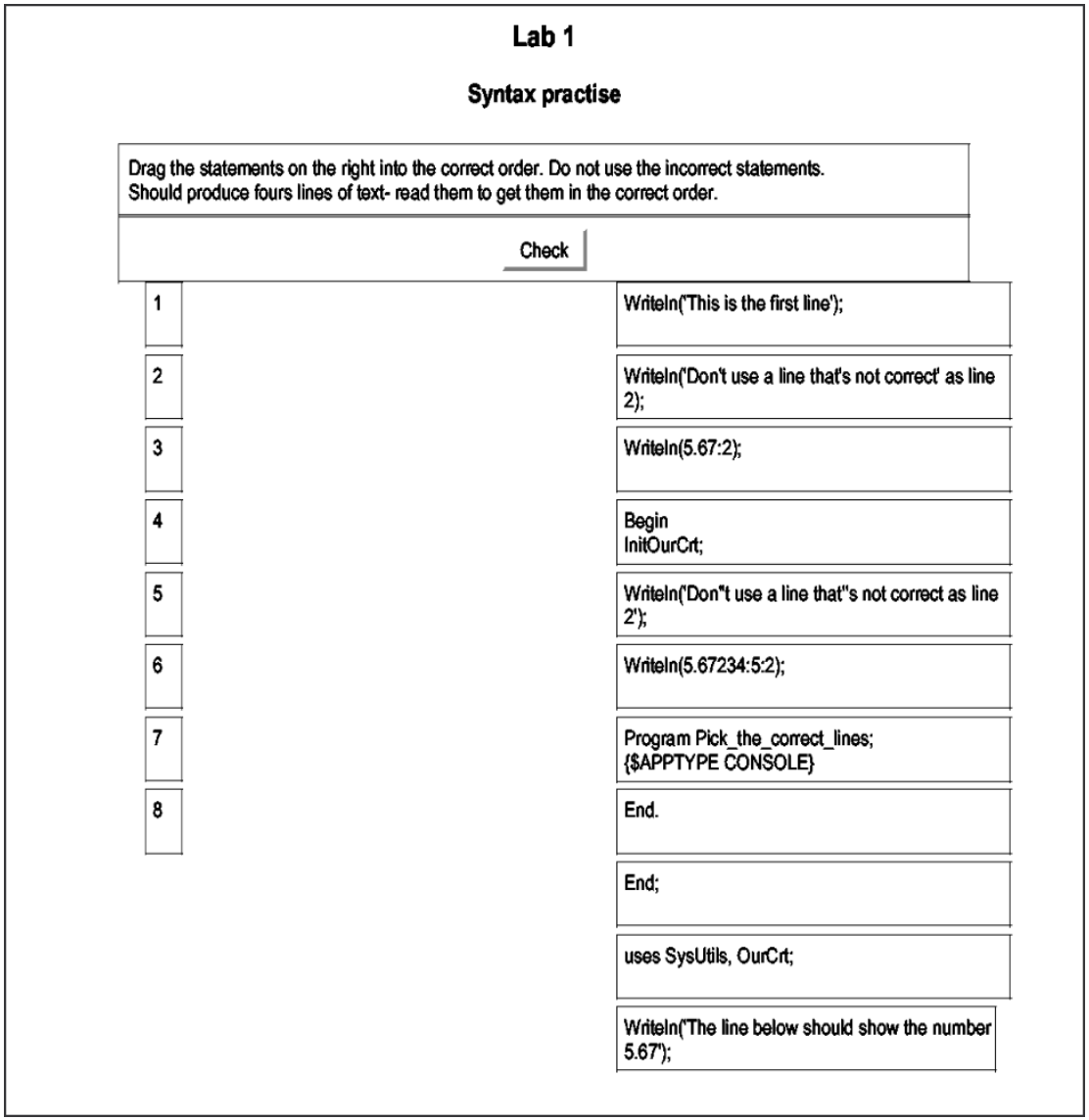


code.org – Hour of Code

Gidget

"Hour of Code," CSEd Week. [Online]. Available: http://csedweek.org/. [Accessed: 18-Mar-2014].
M. J. Lee, F. Bahmani, I. Kwan, J. LaFerte, P. Charters, A. Horvath, F. Luor, J. Cao, C. Law, M. Beswetherick, S. Long, M. Burnett, and A. J. Ko, "Principles of a debugging-first puzzle game for computing education," in 2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC), 2014, pp. 57–64.

More Puzzles…



Lab 1

Syntax practise

Drag the statements on the right into the correct order. Do not use the incorrect statements.
Should produce fours lines of text- read them to get them in the correct order.

Check

1	WriteIn('This is the first line');
2	WriteIn('Don't use a line that's not correct' as line 2);
3	WriteIn(5.67:2);
4	Begin InitOurCrt;
5	WriteIn('Don"t use a line that"s not correct as line 2');
6	WriteIn(5.67234:5:2);
7	Program Pick_the_correct_lines; {$APPTYPE CONSOLE}
8	End.
	End;
	uses SysUtils, OurCrt;
	WriteIn('The line below should show the number 5.67');

Parson's Programming Puzzles

D. Parsons and P. Haden, “Parson’s Programming Puzzles: A Fun and Effective Learning Tool for First Programming Courses,” in Proceedings of the 8th Australasian Conference on Computing Education - Volume 52, Darlinghurst, Australia, Australia, 2006, pp. 157–163.

Learning Programming Independently - Completion Problems

- Generation
  - Write programs from scratch
- Completion
  - Complete partially written programs

J. J. G. Van Merrienboer and M. B. M. De Croock, “Strategies for Computer-Based Programming Instruction: Program Completion Vs. Program Generation,” Journal of Educational Computing Research, vol. 8, no. 3, pp. 365–394, Jan. 1992.

Learning Programming Independently Picture

Completion Problems

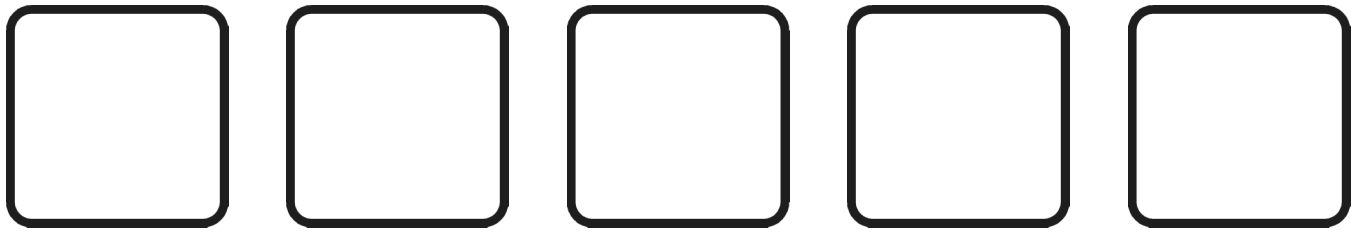
Puzzle-like

Tutorials

How do we effectively use puzzles to support novices learning programming independently?

Cognitive Load Theory

- Completion Problems
- Extraneous Cognitive Load

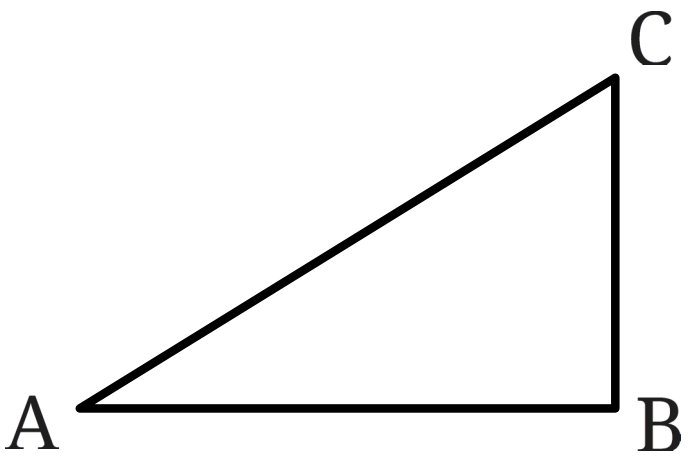


Working Memory

J. Sweller and P. Chandler, “Why Some Material Is Difficult to Learn,” Cognition and Instruction, vol. 12, no. 3, pp. 185–233, 1994.

Extraneous Cognitive Load

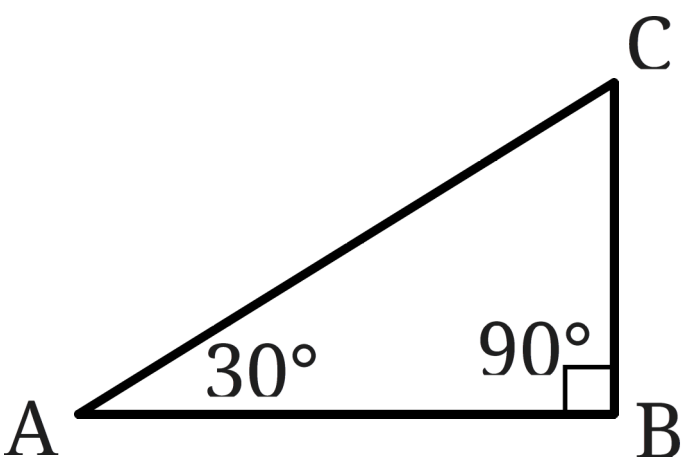
What is angle ACB?

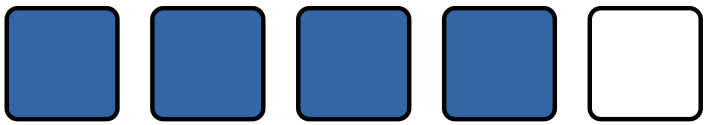


$\angle ABC = 90°$

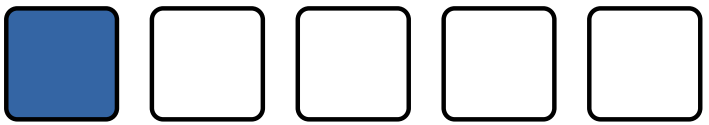
$\angle CAB = 30°$

What is angle ACB?





High Extraneous Cognitive Load



Low Extraneous Cognitive Load

Sweller, P. Ayres, and S. Kalyuga, Cognitive Load Theory. Springer, 2011.

Example: CS1 First Program

HelloWorld.java

```
1
2  public class HelloWorld {
3
4      public static void main(String[] args) {
5
6          System.out.println("Hello World")
7      }
8
9  }
10
```

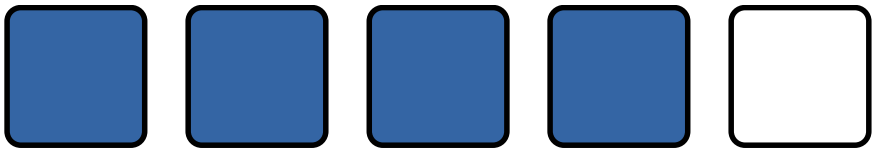
Problems | Javadoc | Declaration

1 error, 0 warnings, 0 others

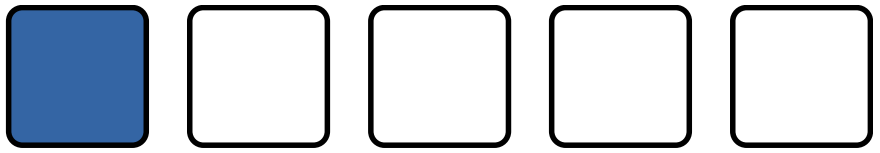
Description

Errors (1 item)

Syntax error, insert ";" to complete BlockStatements

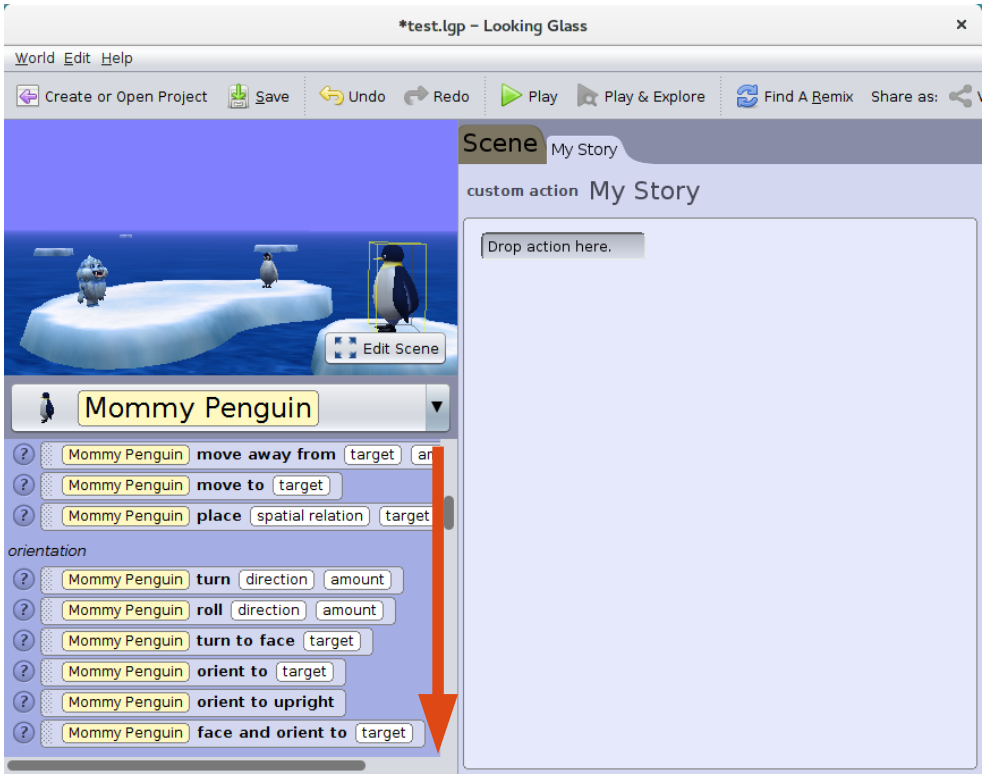


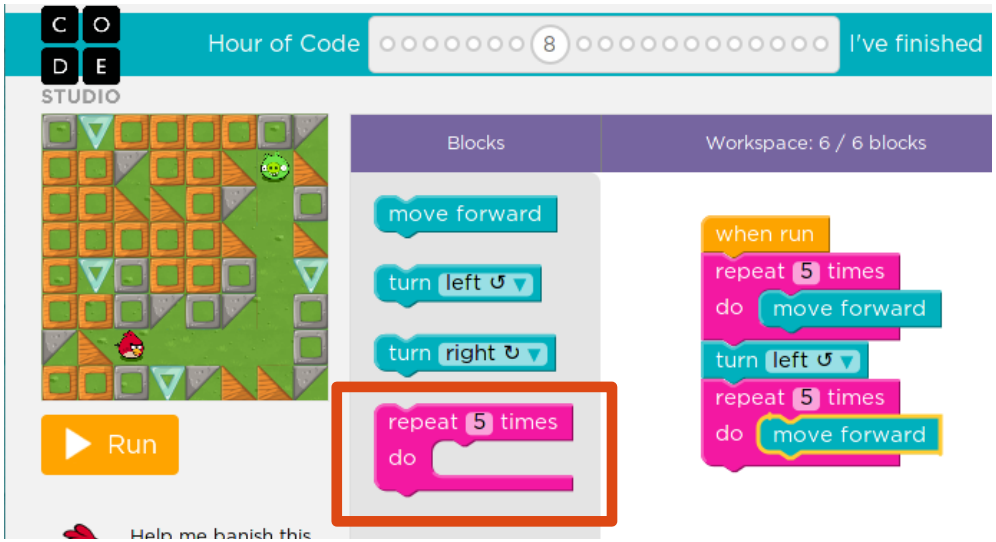
High Extraneous Cognitive Load

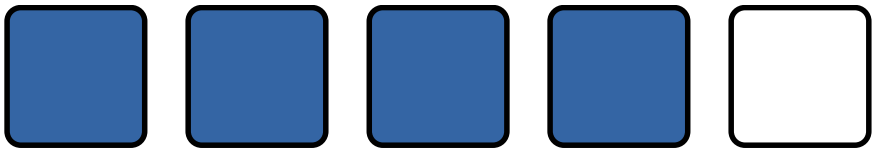


Low Extraneous Cognitive Load

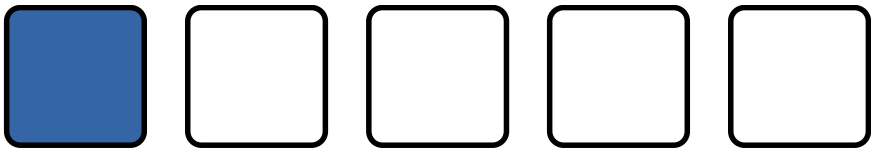
Example: Drag 'n Drop







High Extraneous Cognitive Load



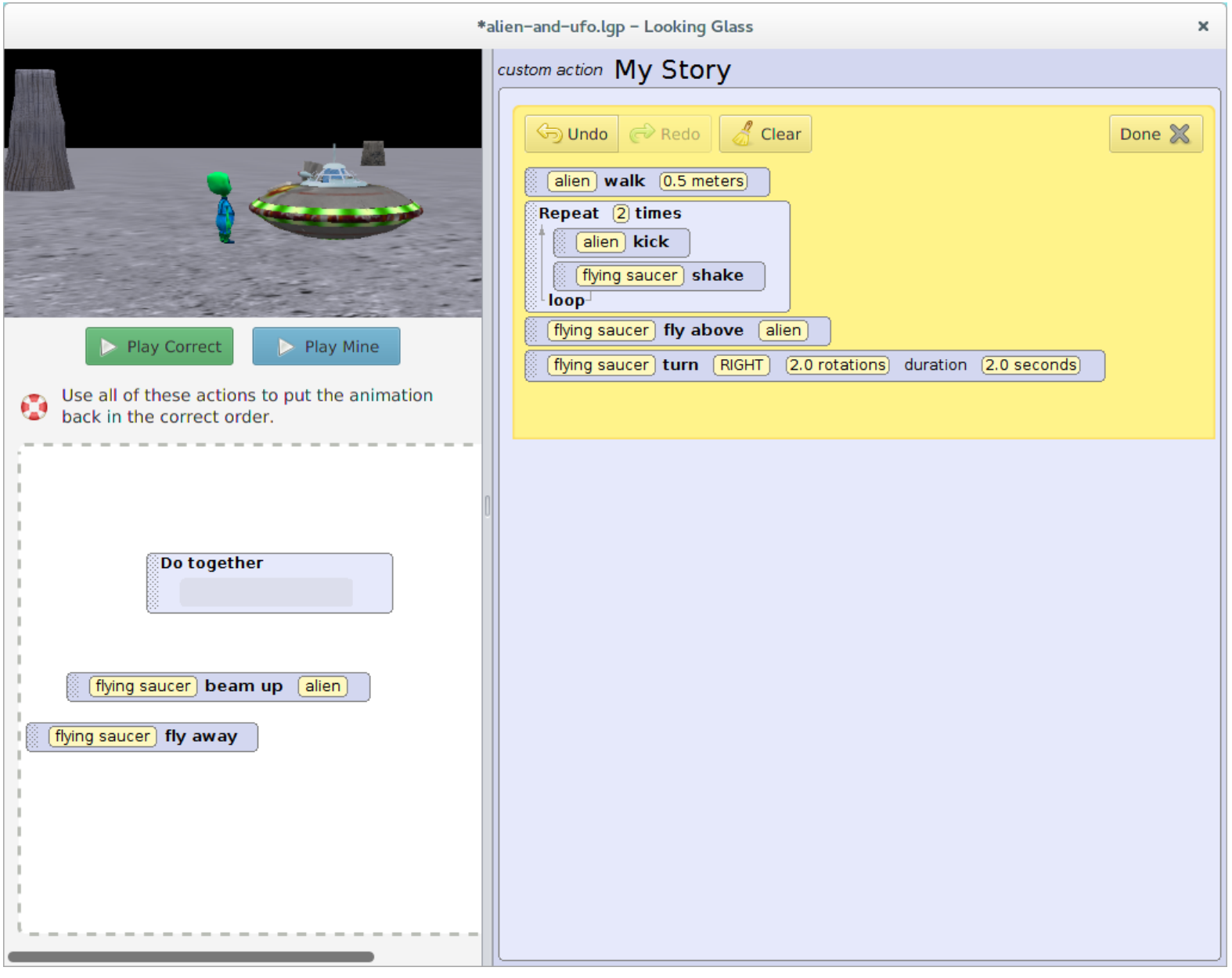
Low Extraneous Cognitive Load

Completion Problems

- Generation (write from scratch)
  - High extraneous cognitive load
- Completion (complete partial program)
  - More working memory resources available for learning

Can we use completion problems and also leverage the strengths of puzzle-like systems to provide an effective way to help novices learn programming independently?

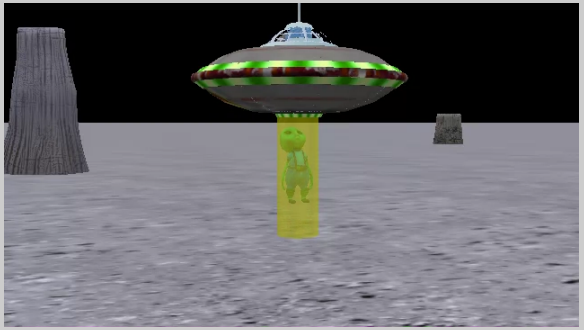
Completion Problems → Programming Completion Puzzles



Puzzle Curriculum

Easy

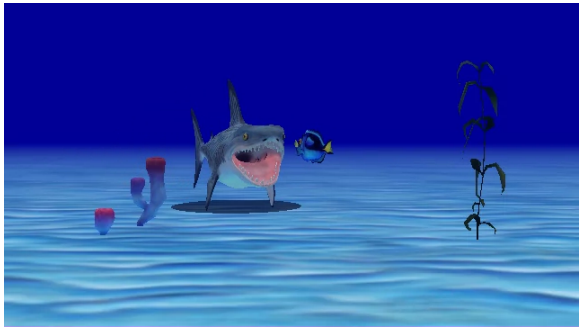
1. Sequential



4. Repeated & Parallel

Challenging

2. Repeated



5. Parallel { Repeated }

3. Parallel

6. Repeated { Parallel }

*alien-and-ufo.lgp – Looking Glass (0)

Play Correct | Play Mine

Use all of these actions to put the animation back in the correct order.

- Repeat 2 times
  - loop
    - flying saucer **fly above** alien
- flying saucer **shake**
- flying saucer **fly away**
- alien **kick**
- flying saucer **beam up** alien
- flying saucer **turn** RIGHT 2.0 rotations du
- **Do together**
  - alien **walk** 0.5 meters

custom action **My Story**

Undo | Redo | Clear | Done

Drop action here.

- **Lessons Learned**
- **Programming Completion Puzzle Effectiveness**

- **Lessons Learned**
- **Programming Completion Puzzle Effectiveness**

Formative Evaluations

- Completion Problem → Puzzle Format & Interface
  - 10 iterations
  - 23 participants - St. Louis Science Center
  - 30 minutes
- Puzzle Curriculum
  - 8 iterations
  - 21 participants - St. Louis Academy of Science
  - 90 minutes

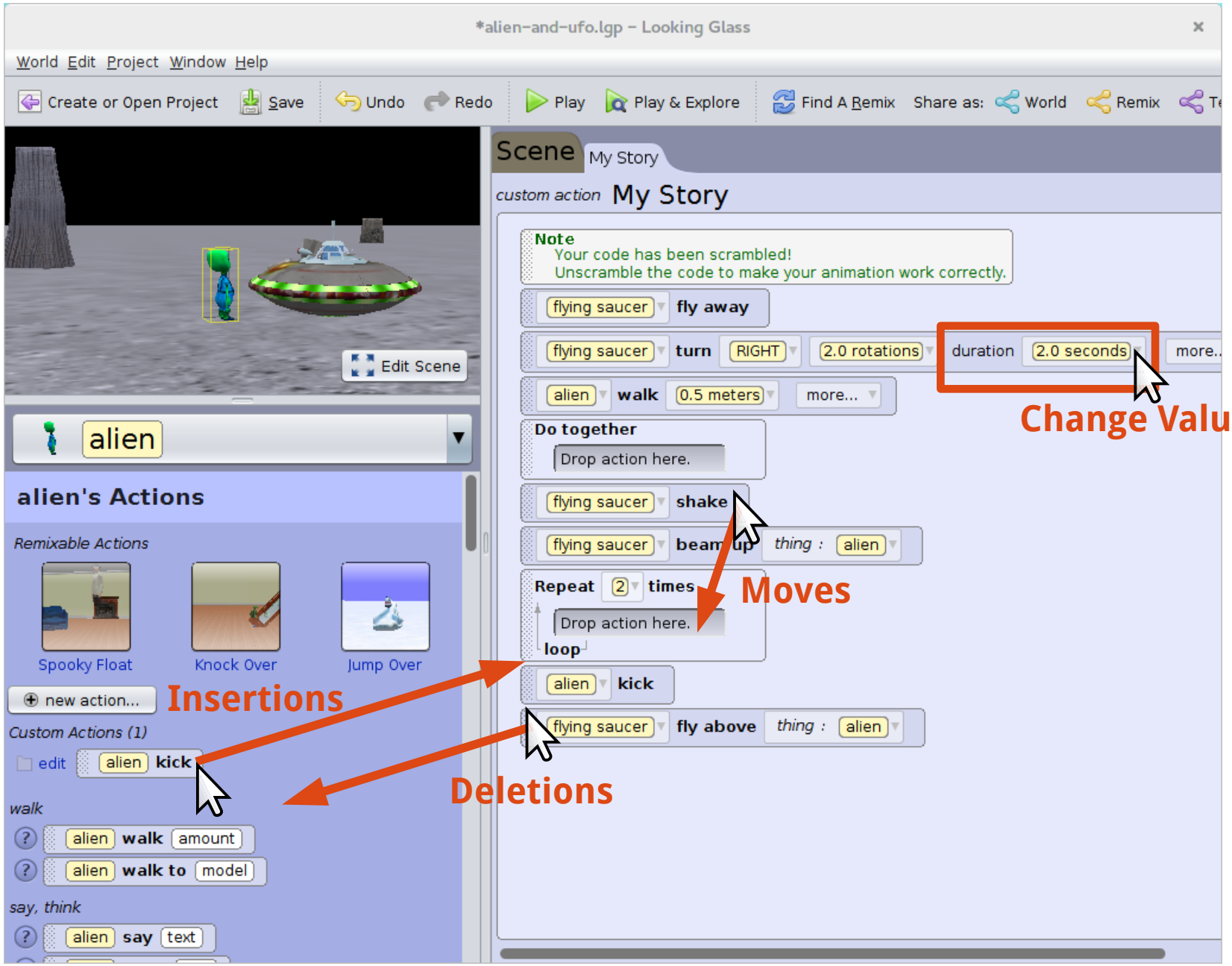
Lessons Learned:

1) Limit the editable dimensions of the puzzle.
2) When executing the program, limit distractions and focus the user's attention on the program's output.
3) Author puzzle programs with memorable segments.
4) Provide a challenge without being tricky.

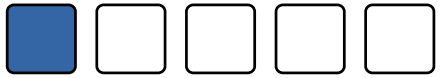
...

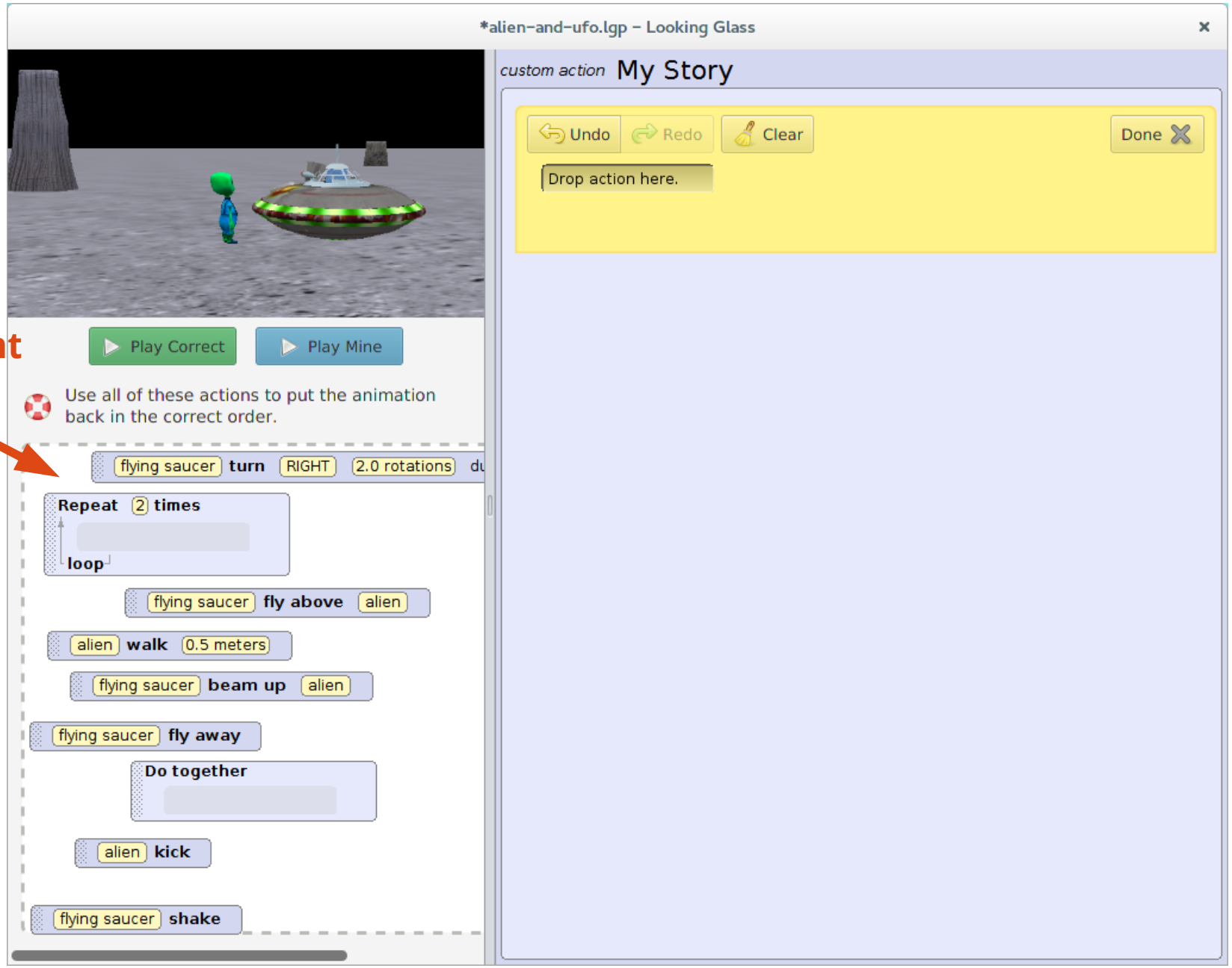
1) Limit the editable dimensions of the puzzle.

Editable Dimensions



Limit the Possibilities

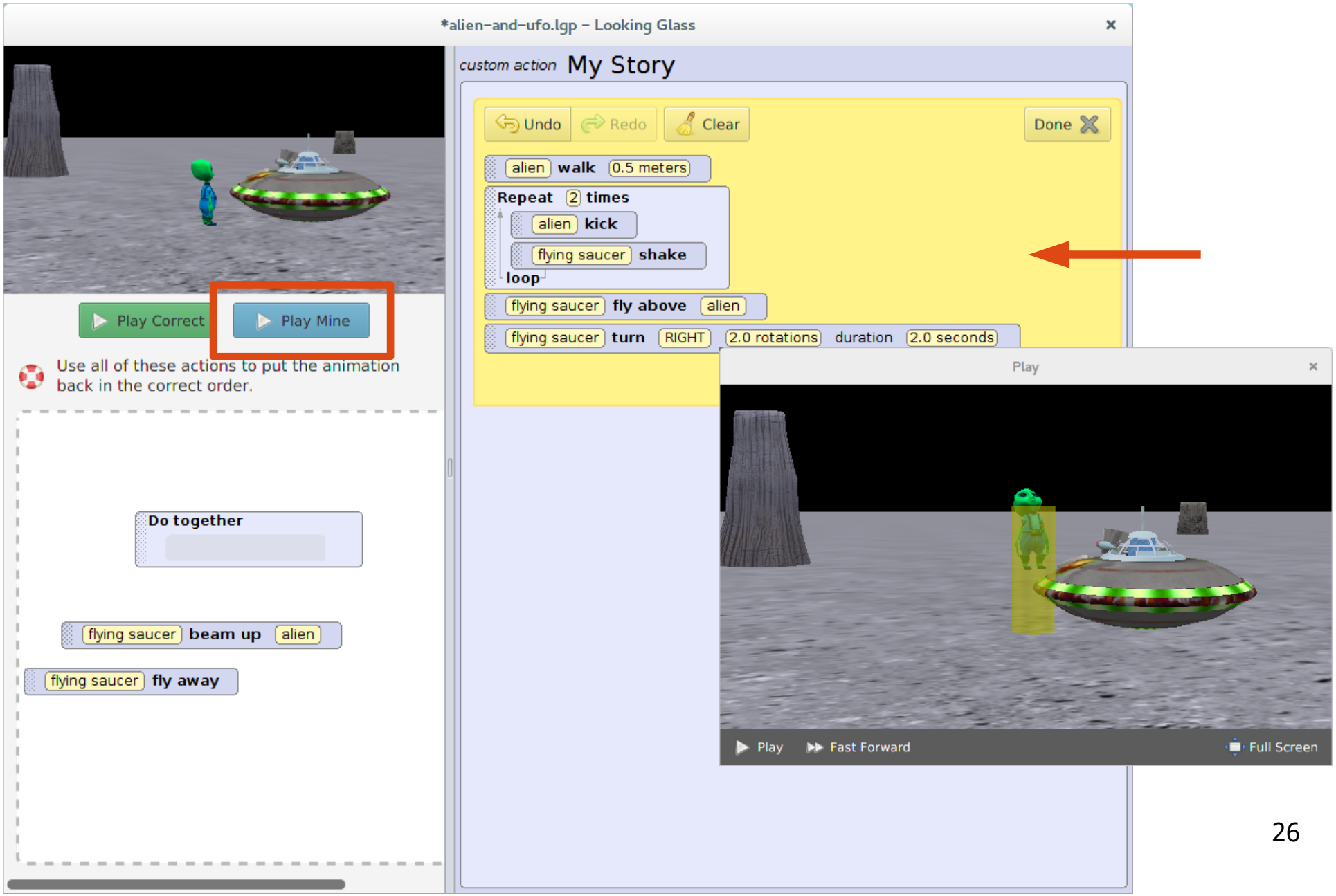




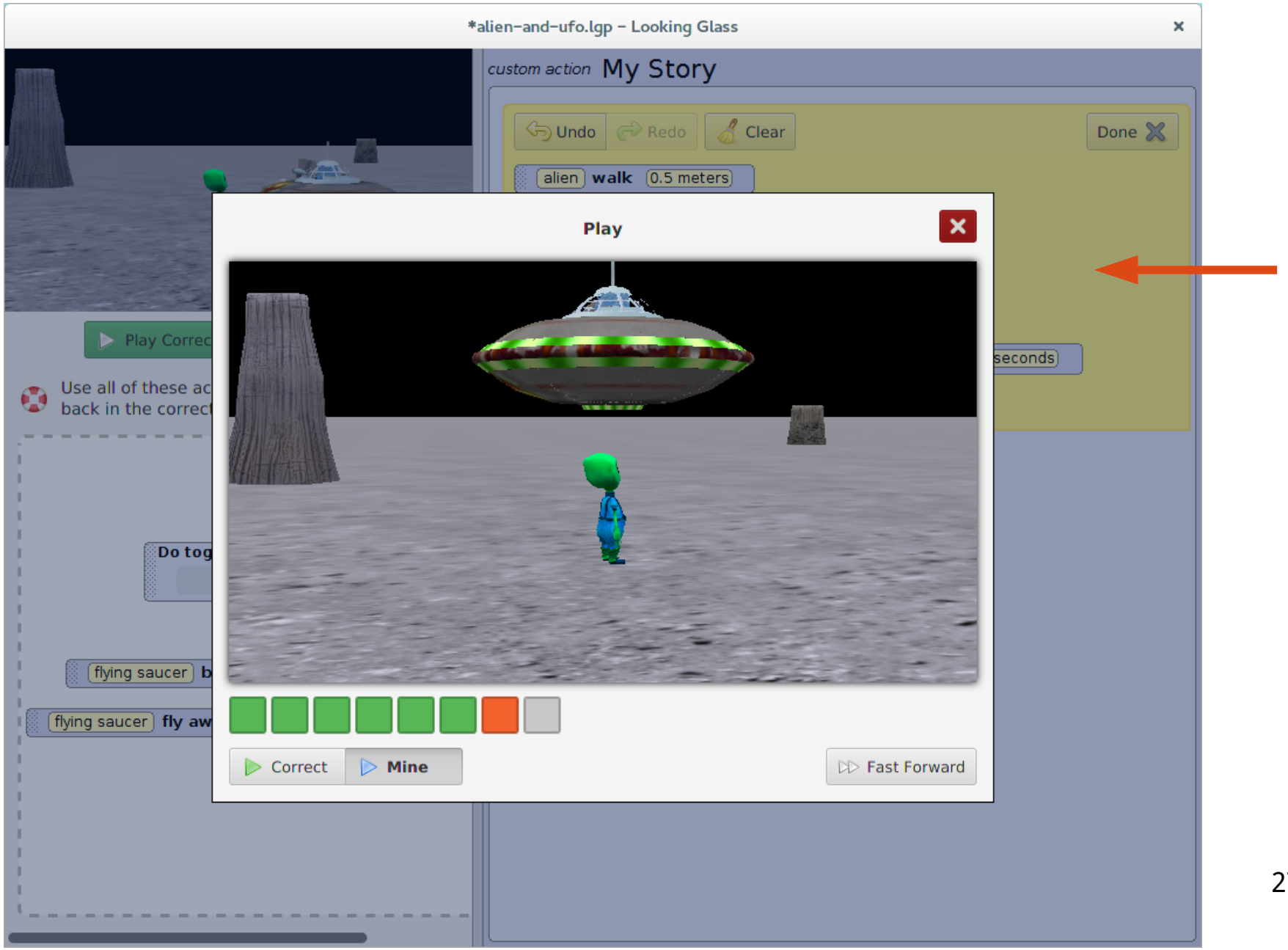
Statement Bin

2) When executing the program, limit distractions and focus the user's attention on the program's output.

Play Window


*alien-and-ufo.lgp – Looking Glass
Play Correct
Play Mine
Use all of these actions to put the animation back in the correct order.
Do together
flying saucer beam up alien
flying saucer fly away
custom action My Story
Undo
Redo
Clear
Done
alien walk 0.5 meters
Repeat 2 times
alien kick
flying saucer shake
loop
flying saucer fly above alien
flying saucer turn RIGHT 2.0 rotations duration 2.0 seconds
Play
Play
Fast Forward
Full Screen

Play Overlay



3) Author puzzle programs with memorable segments.

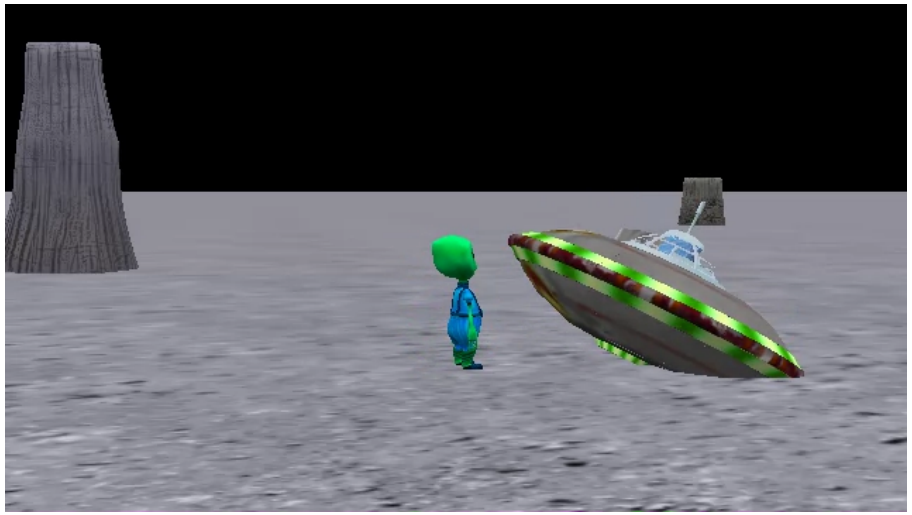
Hard to Remember Output

Monkey Business

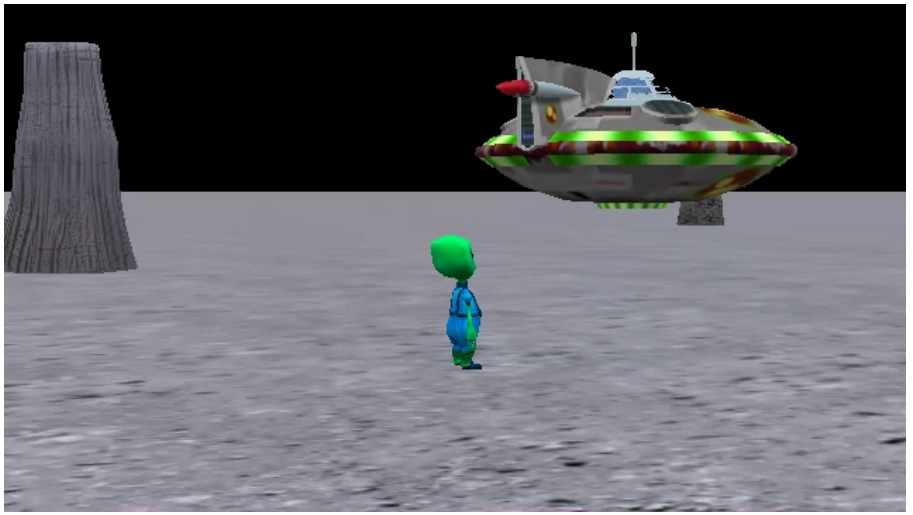
```
Brown Monkey turn to face Grey Monkey
Grey Monkey turn to face Brown Monkey
Repeat 2 times
    Brown Monkey talk
    Grey Monkey talk
loop
Brown Monkey move UP 0.5 meters duration 0.5 seconds animationStyle BEGIN ABRUPTLY AND END GENTLY
Brown Monkey move DOWN 0.5 meters duration 0.5 seconds animationStyle BEGIN GENTLY AND END ABRUPTLY
Repeat 3 times
    Grey Monkey say "EEK EEK"
    Brown Monkey turn RIGHT 4.0 rotations
    Grey Monkey hop side to side
loop
Grey Monkey lay down
```

Monkey Business

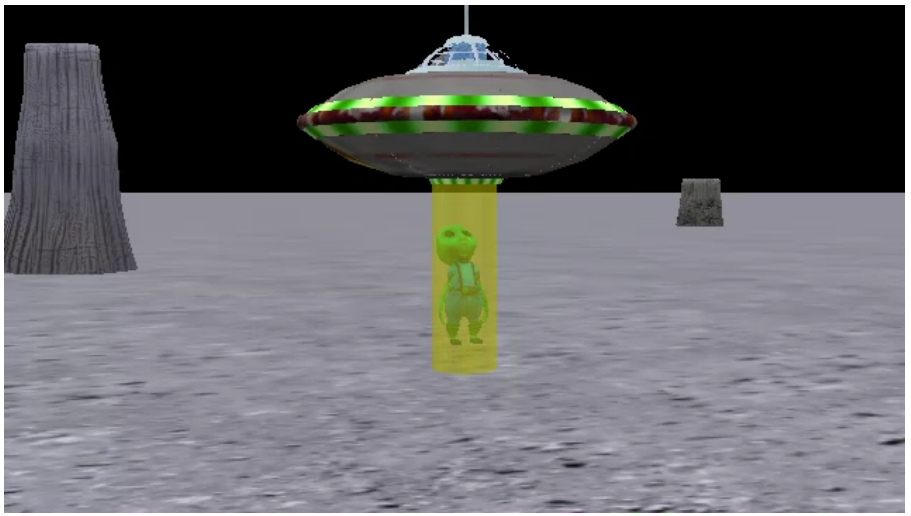
Memorable Segments



Segment I – The alien repairs the flying saucer.

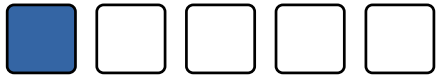


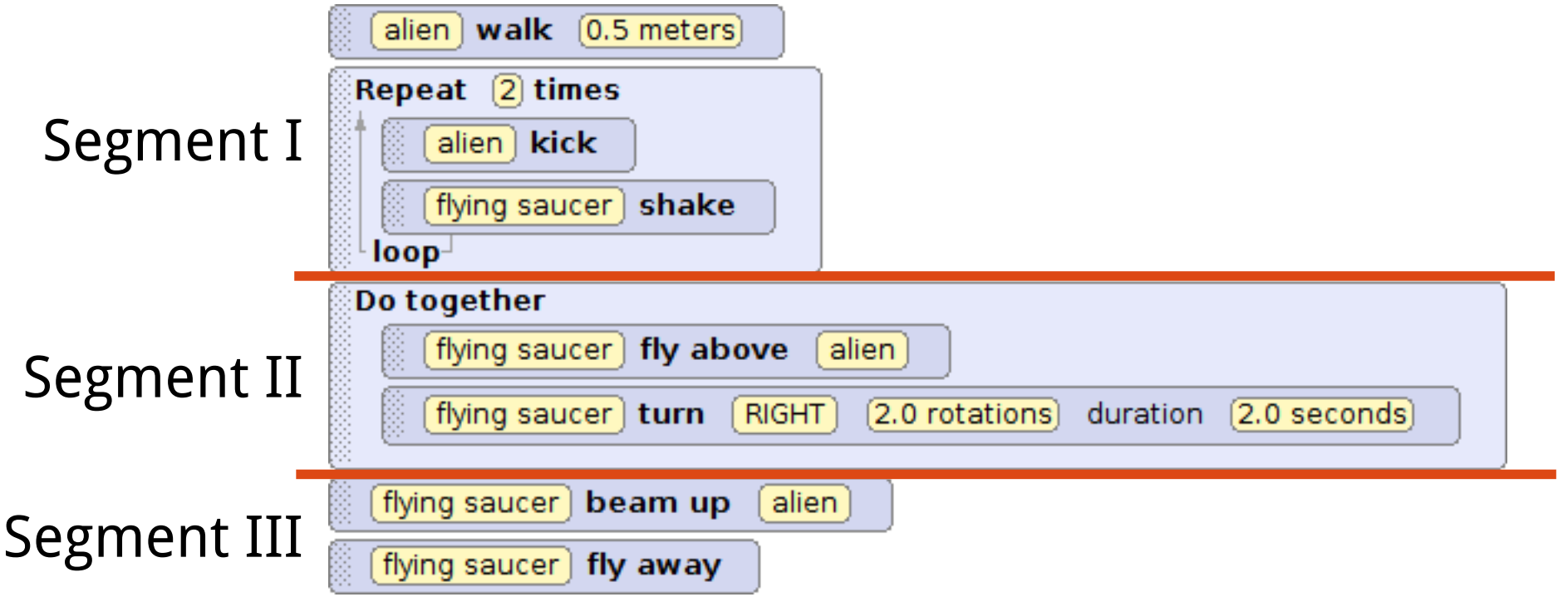
Segment II – The flying saucer starts up.



Segment III - Alien drives flying saucer.

Memorable Segments





4) Provide a challenge without being tricky.

Tricky & Challenging

"It was tricky, but not harder."

"I thought this one was a little challenging, but I liked it!"

Tricky

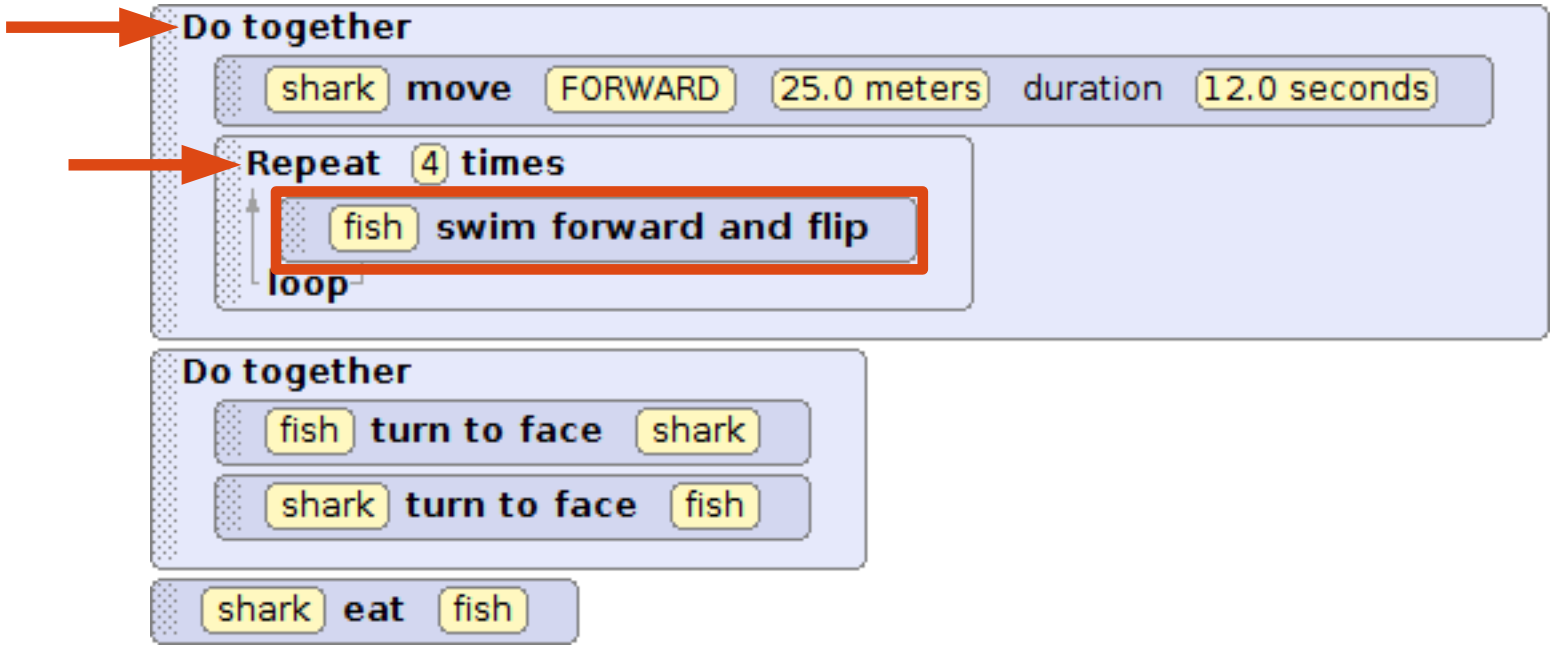
Nearly Identical Statements

witch **move** FORWARD 1.0 meters
witch **move** FORWARD 1.2 meters

witch **move** BACKWARD 1.0 meters animationStyle BEGIN GENTLY AND END ABRUPTLY
witch **move** BACKWARD 1.0 meters animationStyle BEGIN AND END GENTLY

witch **move** FORWARD 1.0 meters duration 2.0 seconds
witch **move** FORWARD 1.0 meters duration 1.5 seconds

Challenging

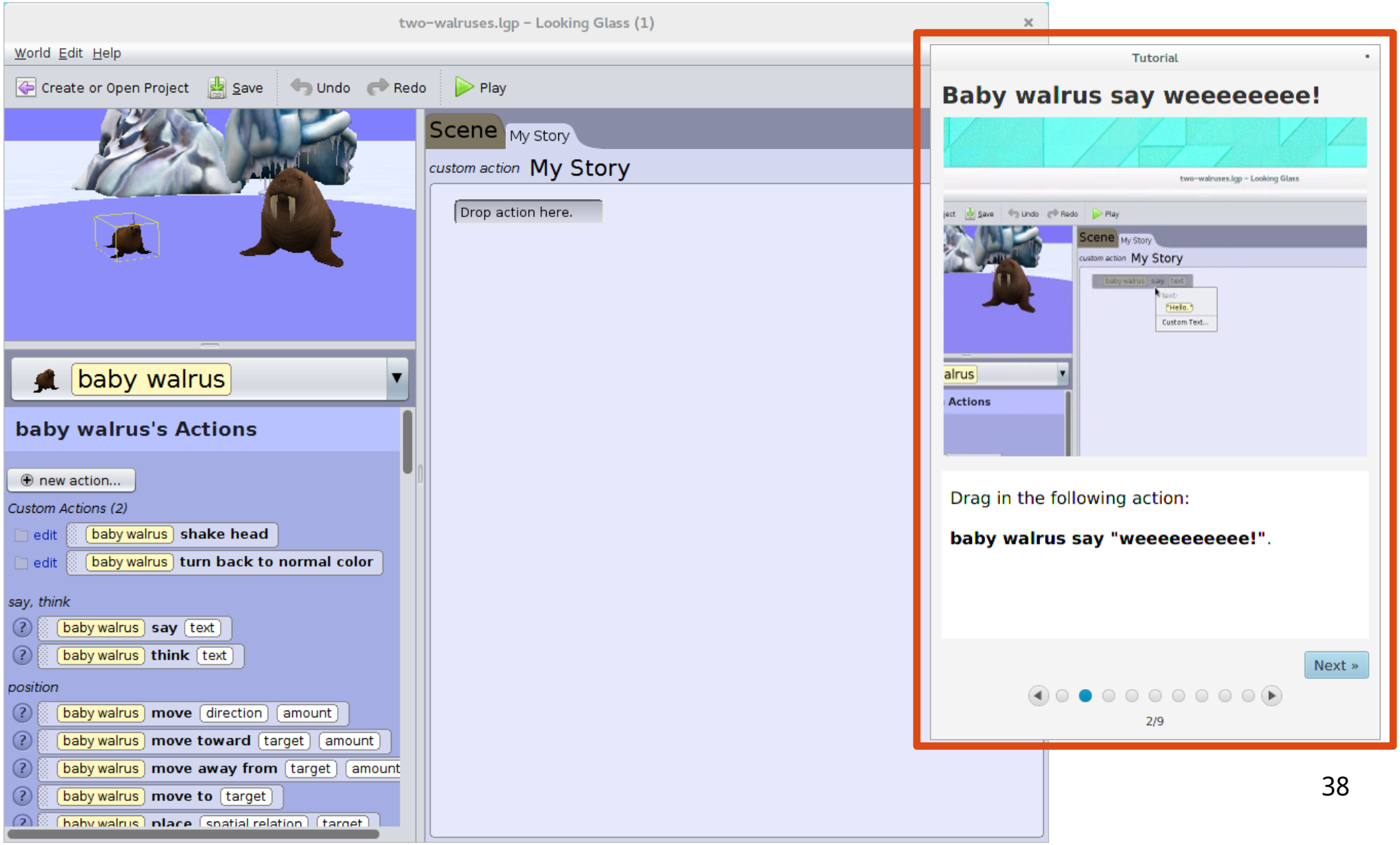


- Lessons Learned
- **Programming Completion Puzzle Effectiveness**

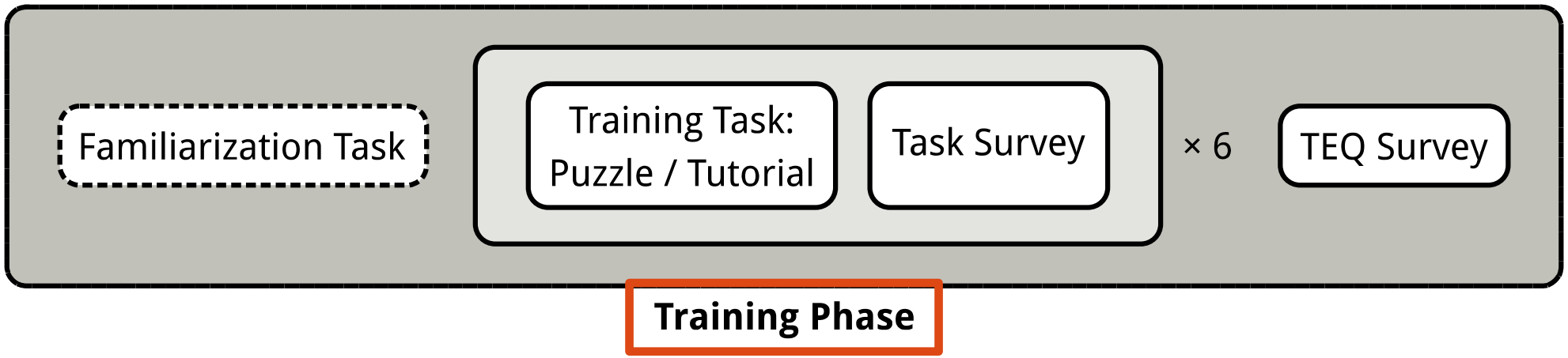
Summative Evaluation

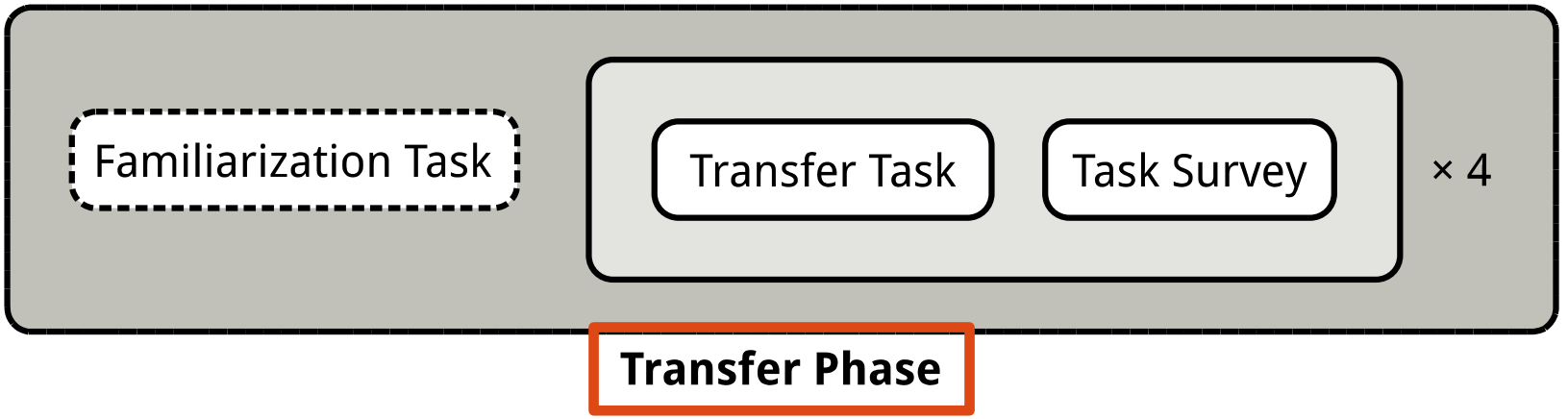
- 27 participants
  - 12 Female, 15 Male
  - Average Age: 11.59
  - Minimal Programming Experience (< 3 hours)
- 2 hours

Independent Learning: Tutorials

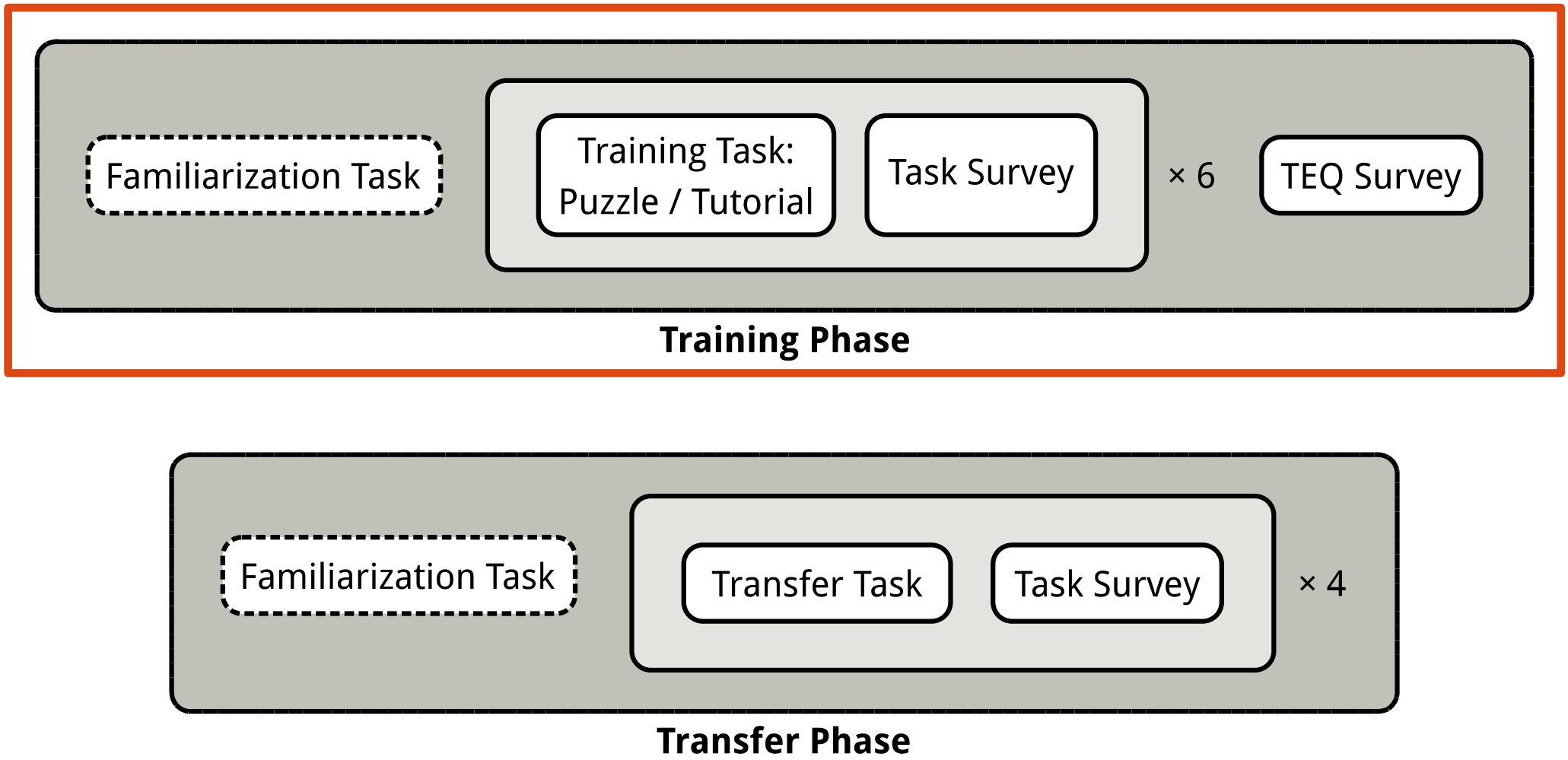


Study Design





Study Design

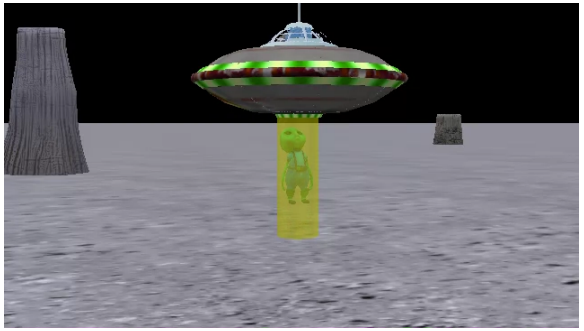


Training Phase

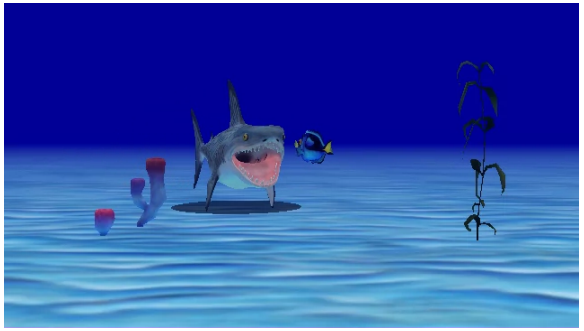
1. Sequential

2. Repeated

3. Parallel



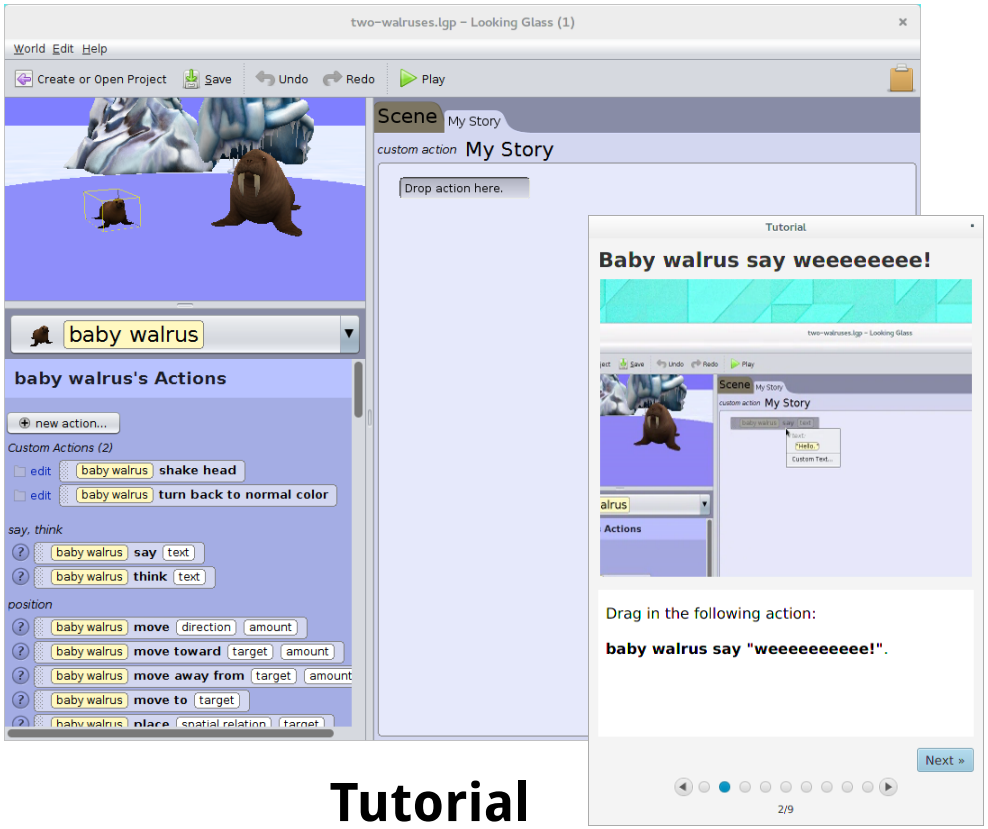
4. Repeated &
Parallel



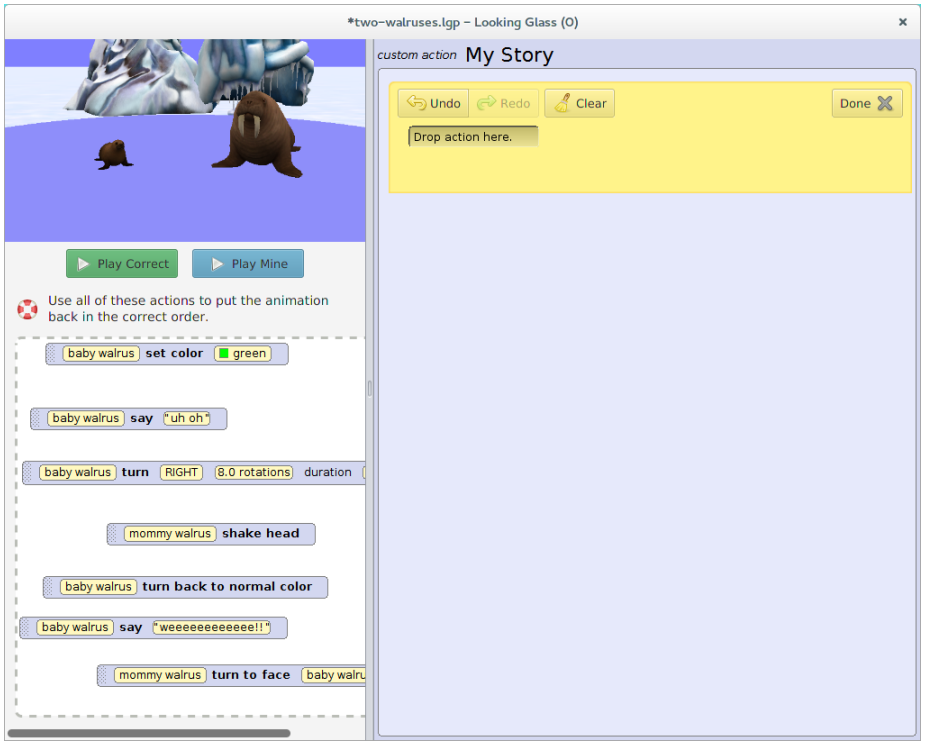
5. Parallel {
Repeated
}

6. Repeated {
Parallel
}

Training Task

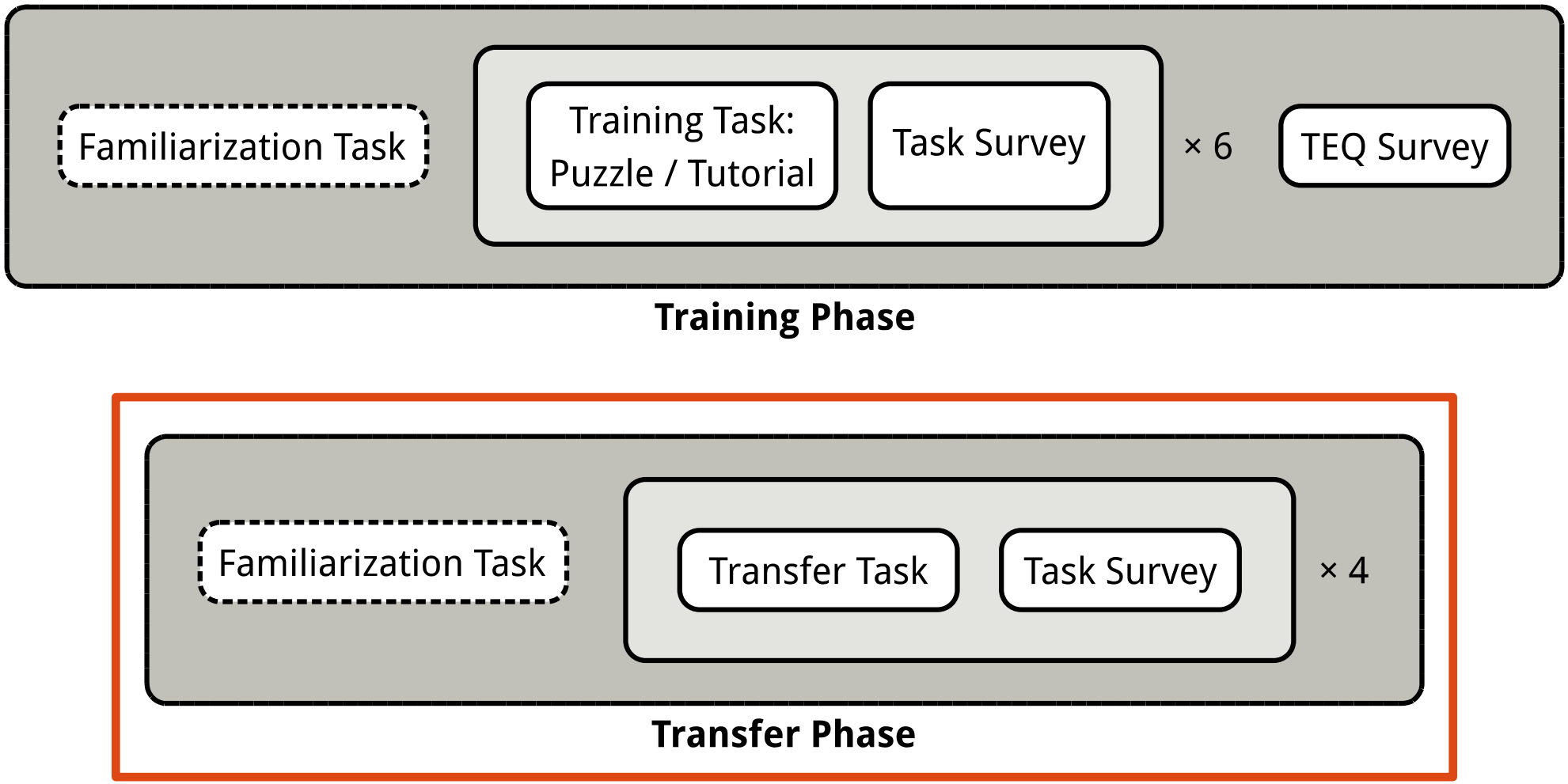


Tutorial



Puzzle

Study Design



Transfer Phase

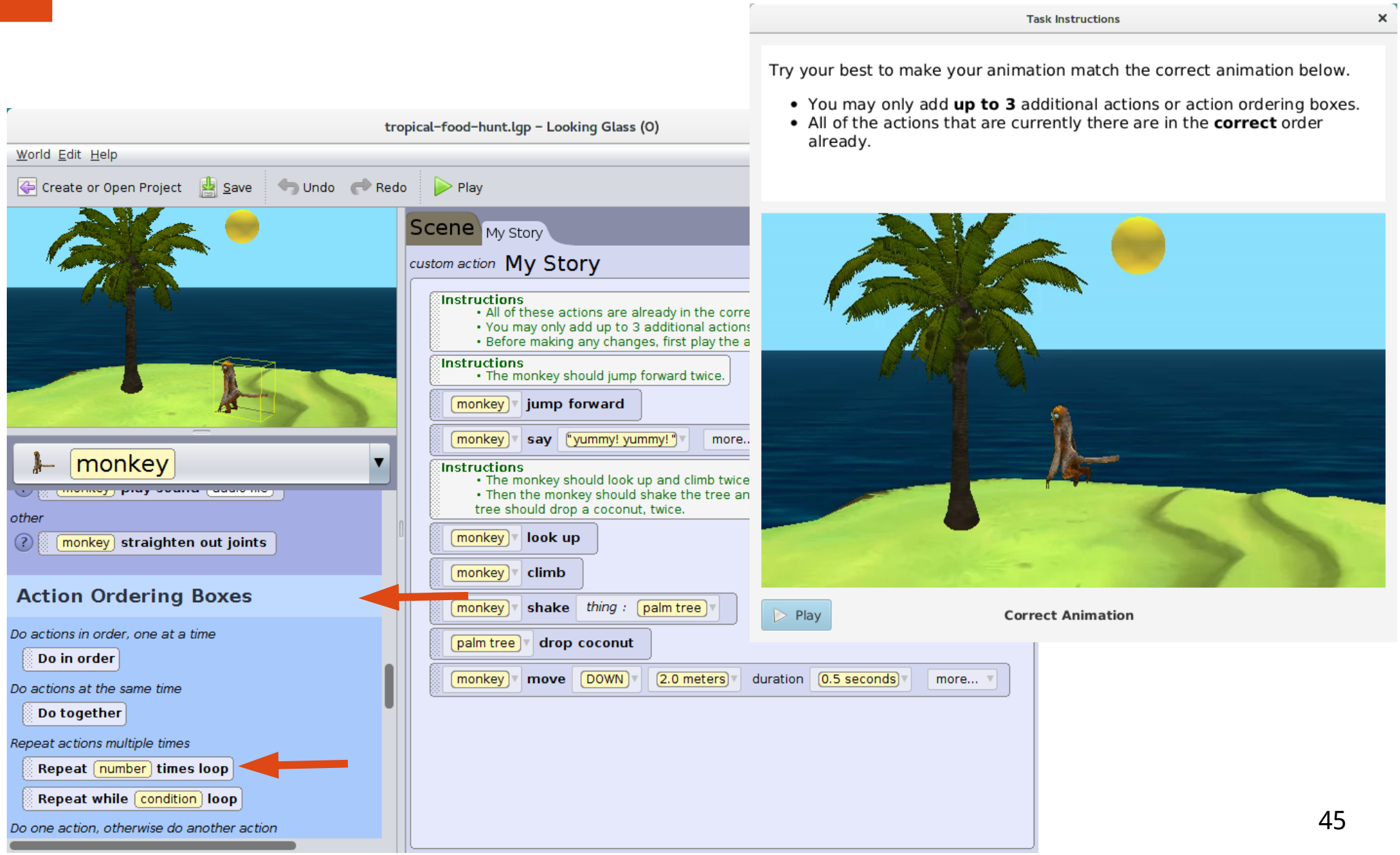
Repeated

Parallel

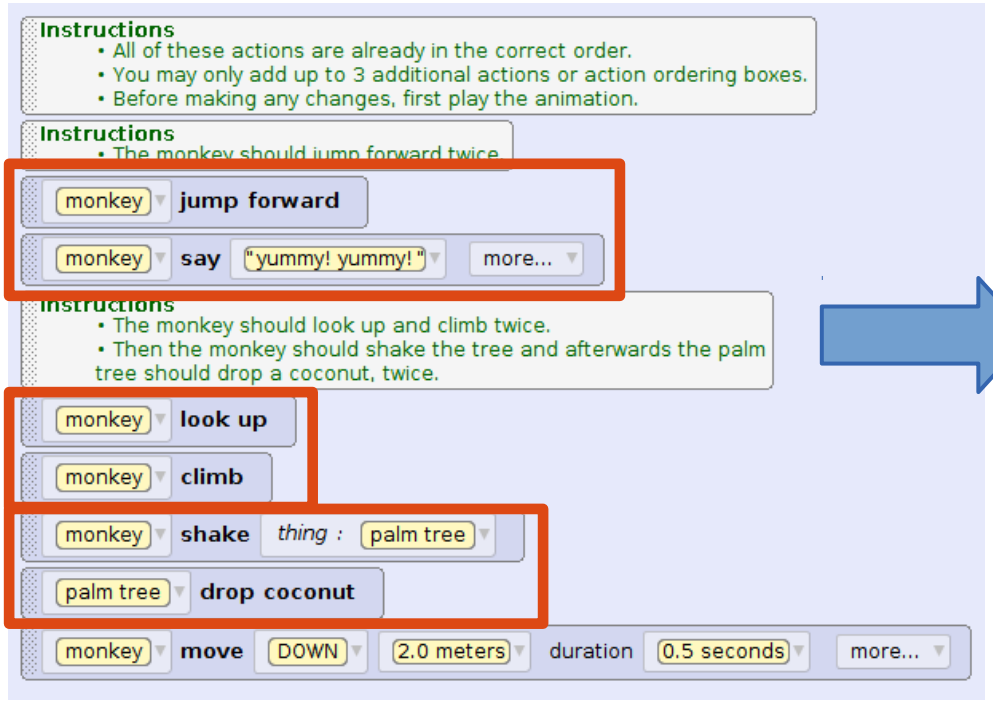
Parallel {
Repeated
}

Repeated {
Parallel
}

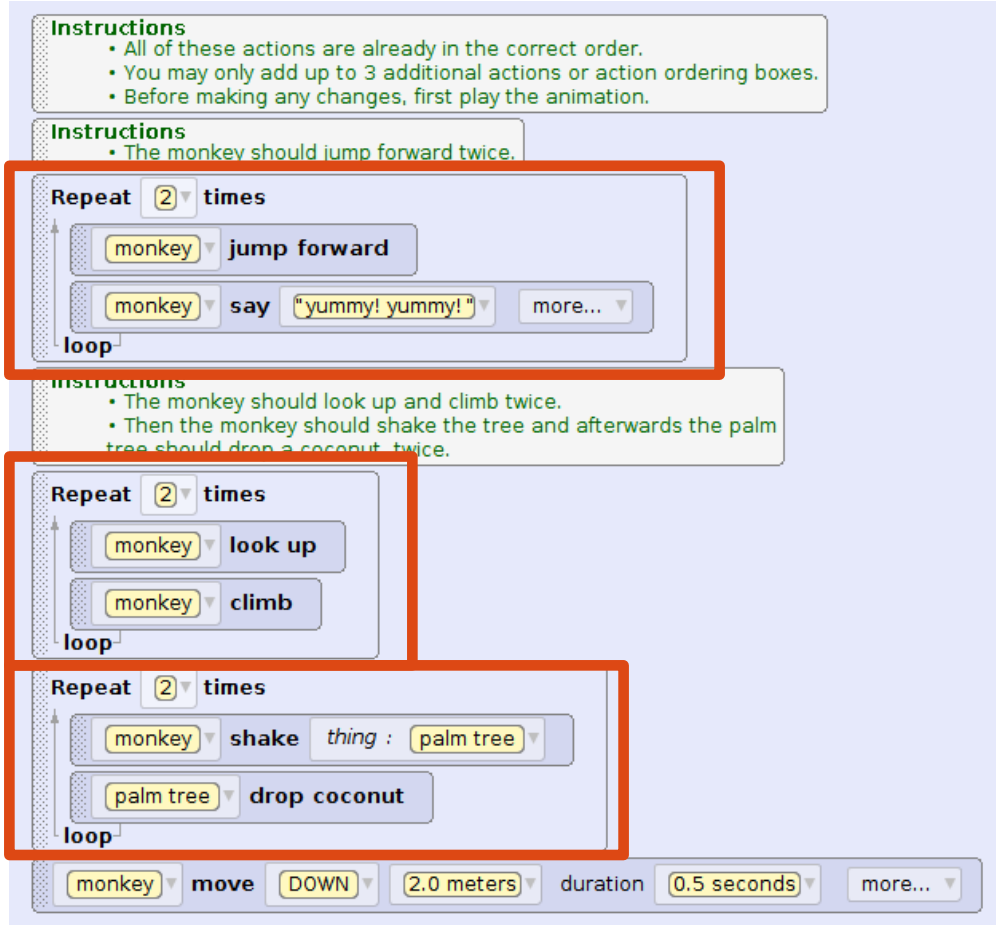
Transfer Task



Completed Transfer Task



Initial



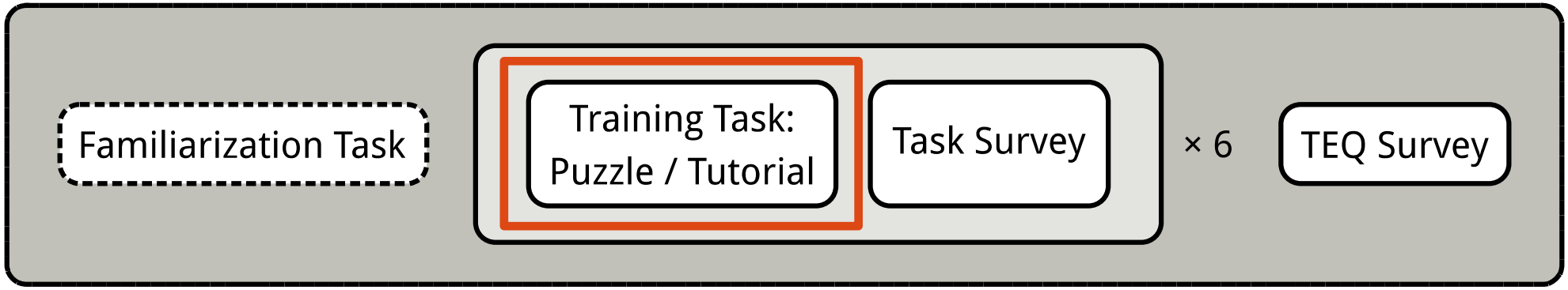
Completed

Research Questions:

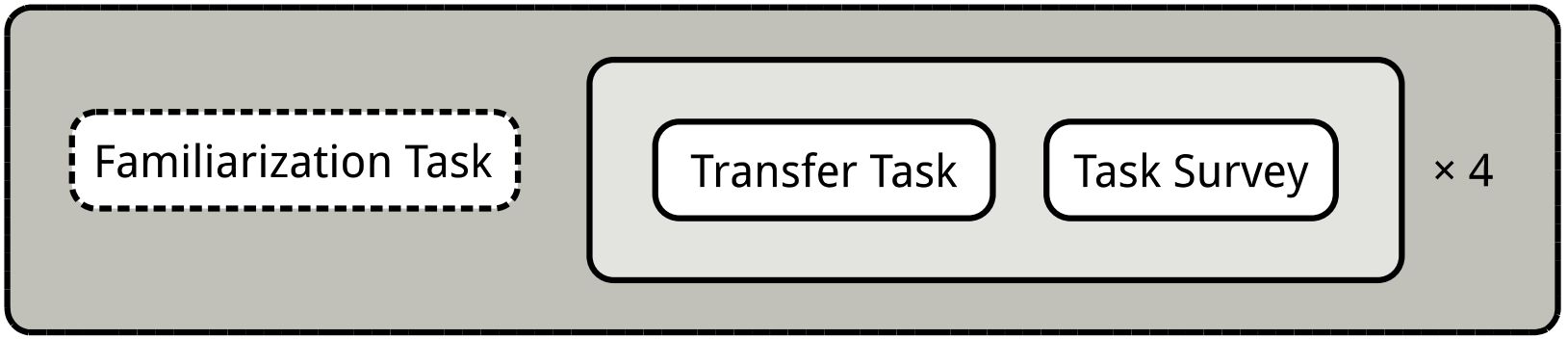
1) Do puzzles require a different time and mental investment compared to tutorials?
2) Do puzzle users show more evidence of learning compared to tutorial users?
3) Are puzzles more motivating than tutorials?

1) **Do puzzles require a different time and mental investment compared to tutorials?**

Study Design

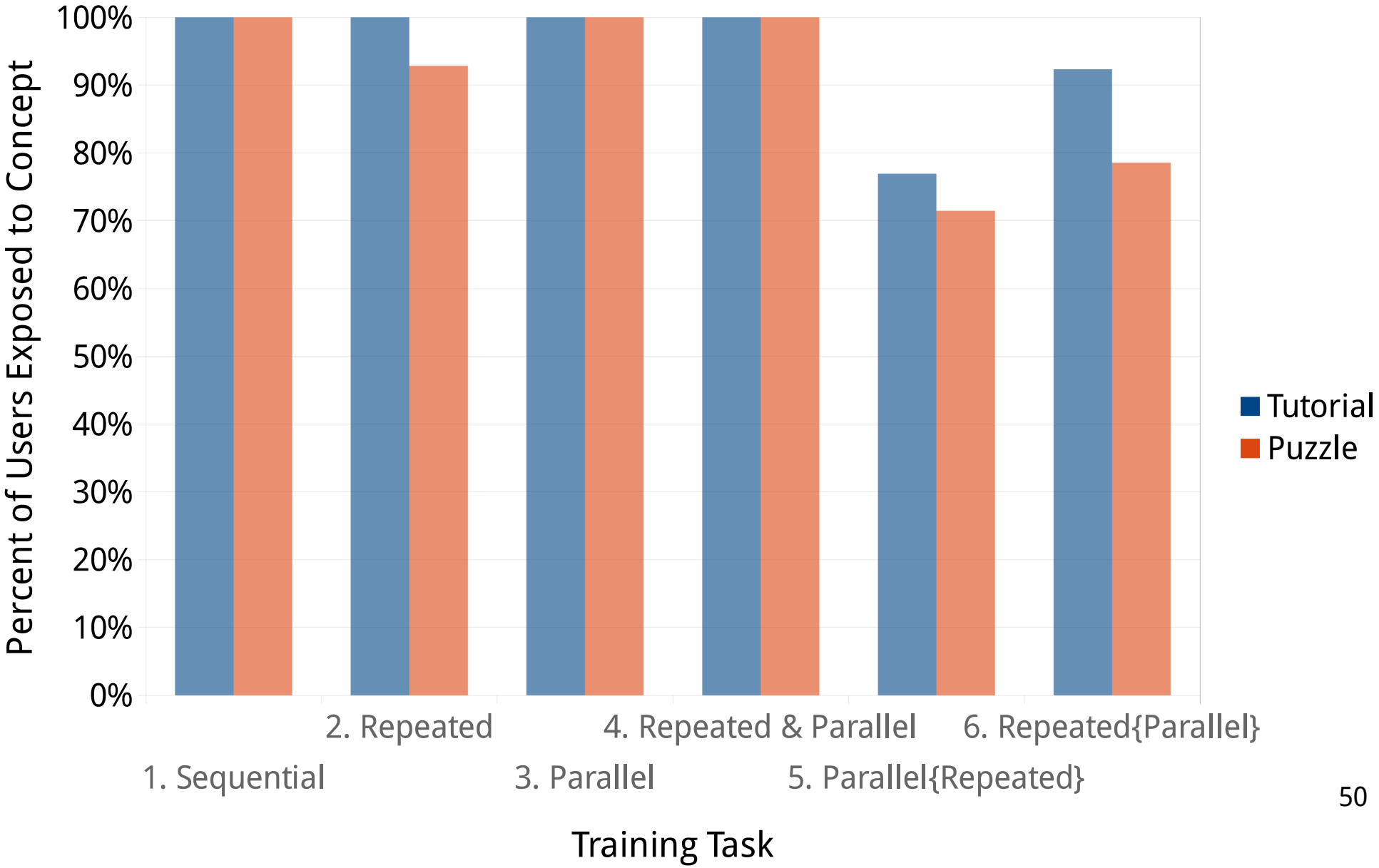


Training Phase



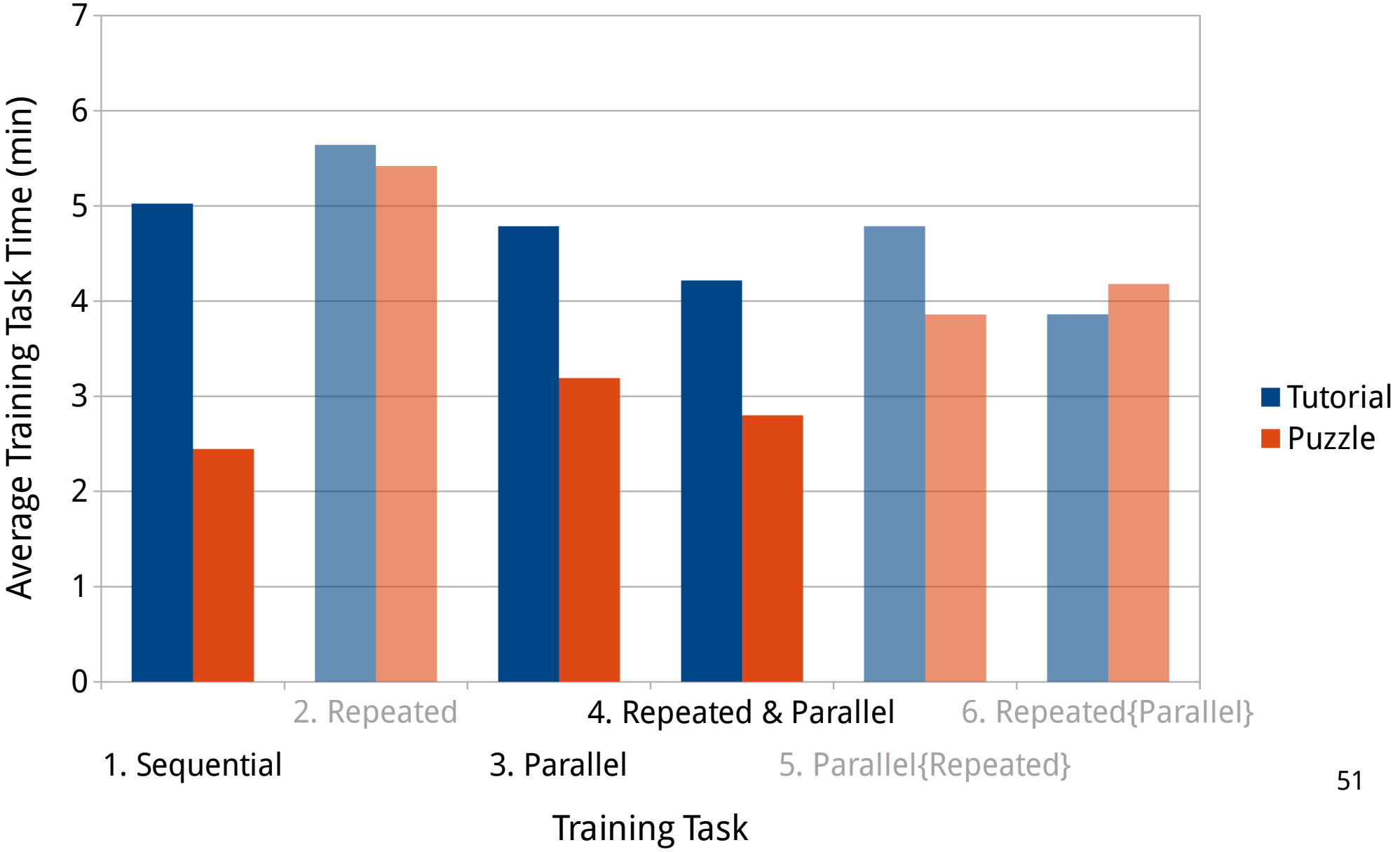
Transfer Phase

Percent of Users Exposed to Programming Concept



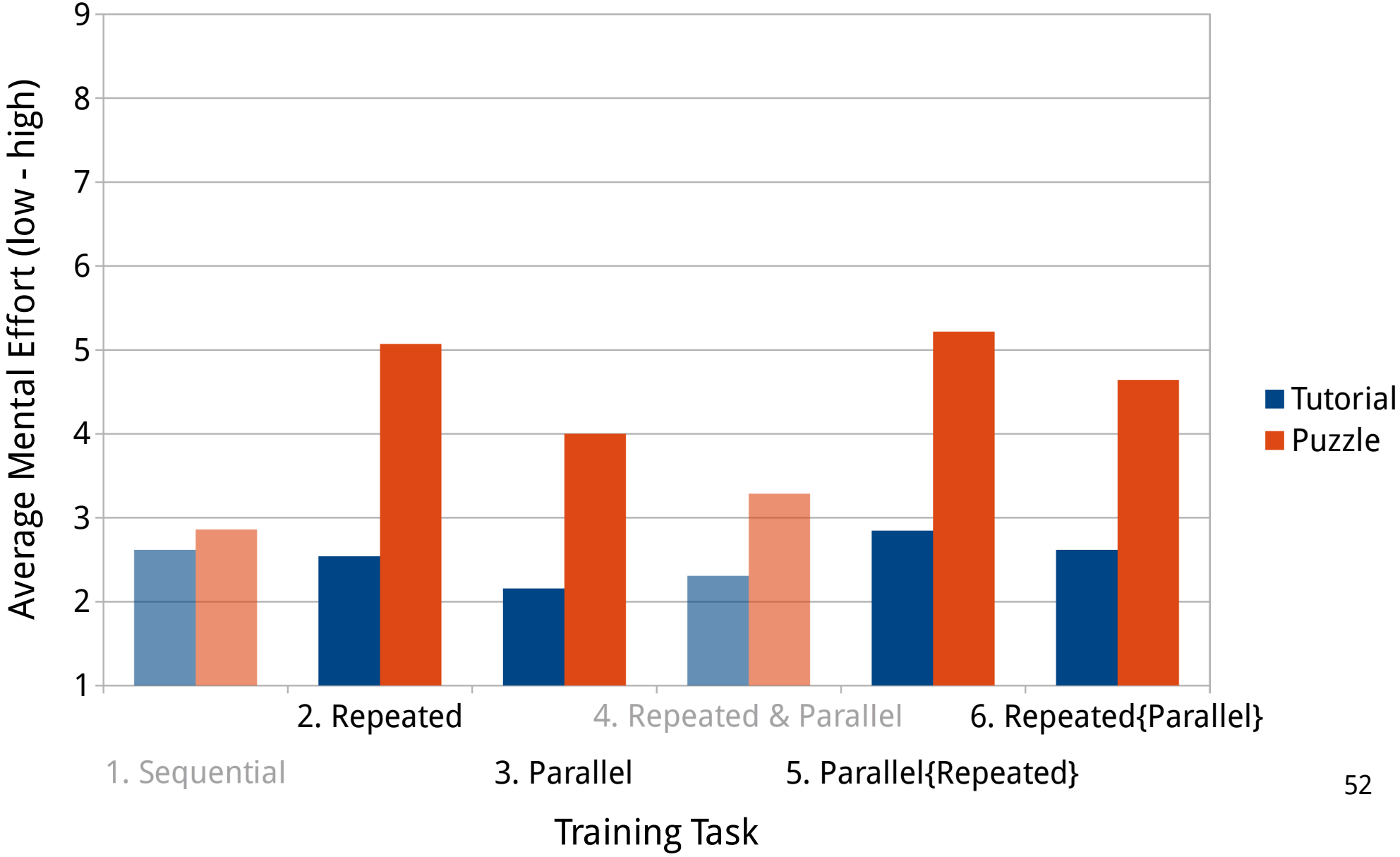
Average Training Task Time

$p < .001$



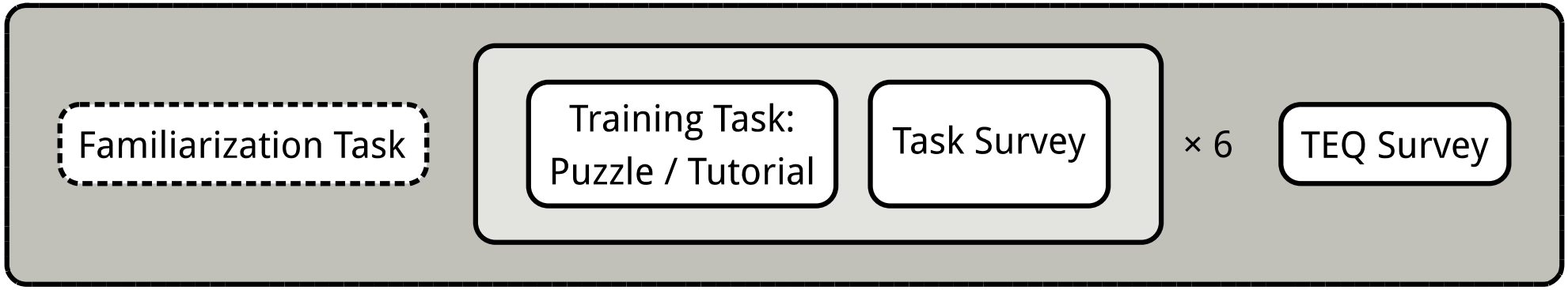
Average Training Task Mental Effort

$p < .05$

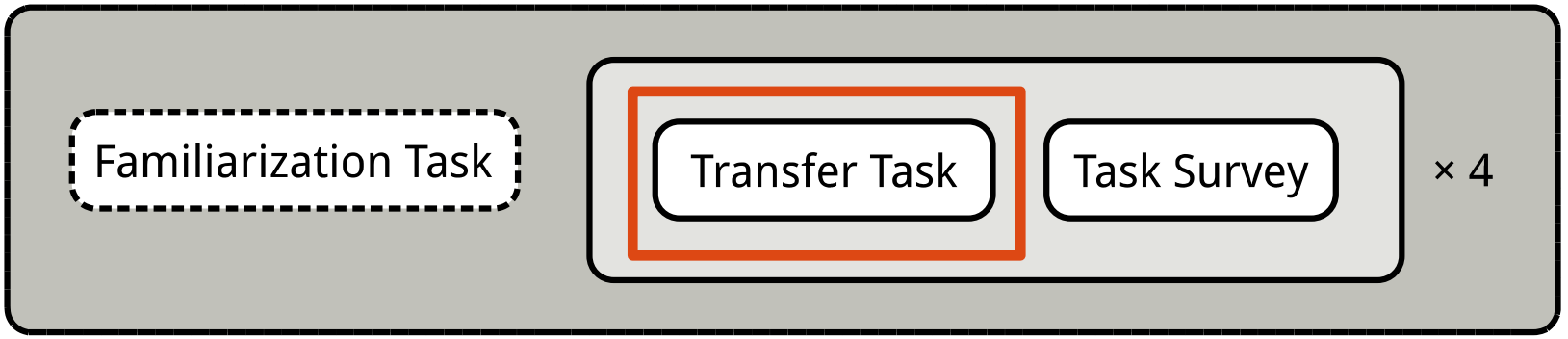


2) Do puzzle users show more evidence of learning compared to tutorial users?

Study Design



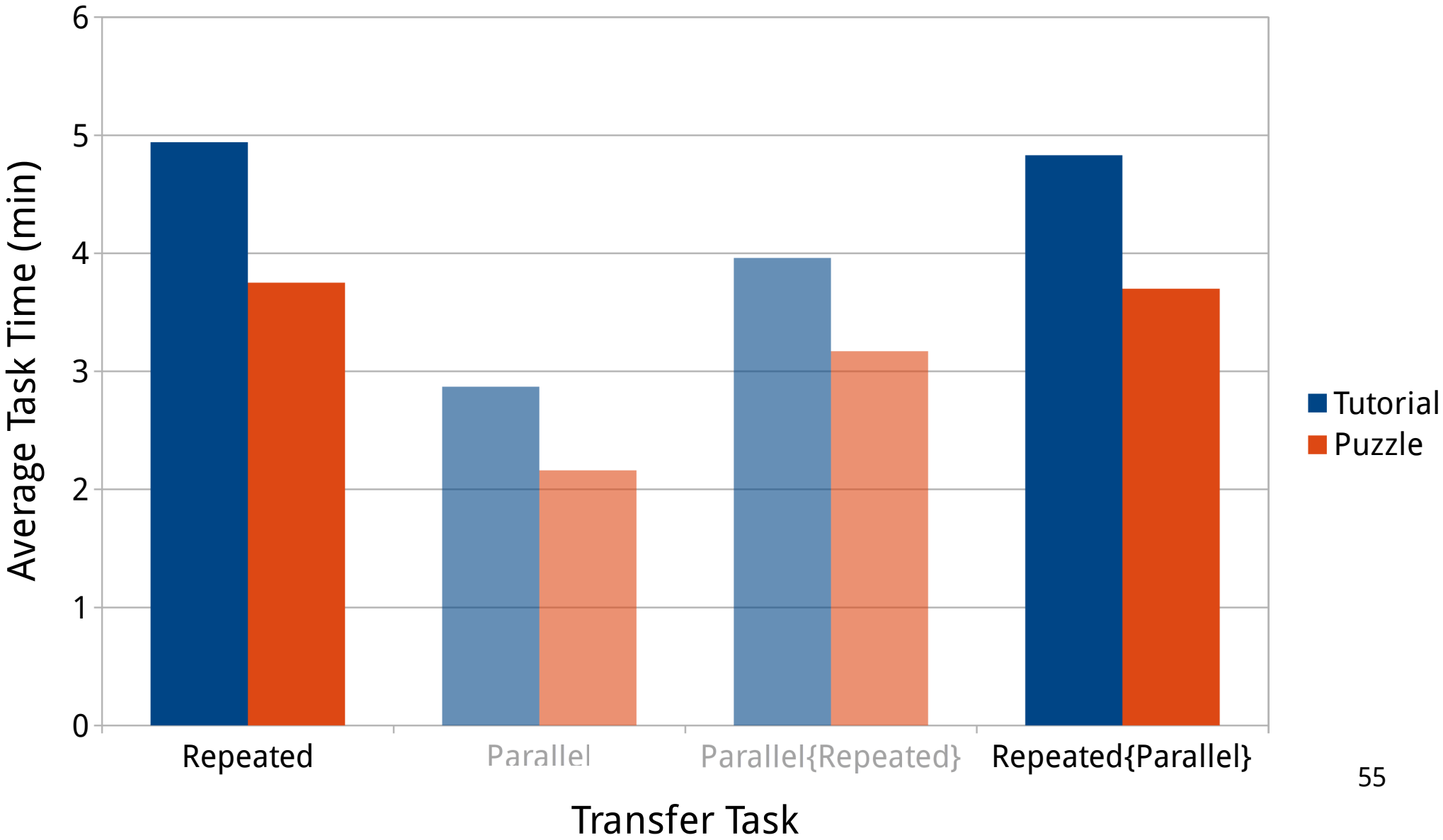
Training Phase



Transfer Phase

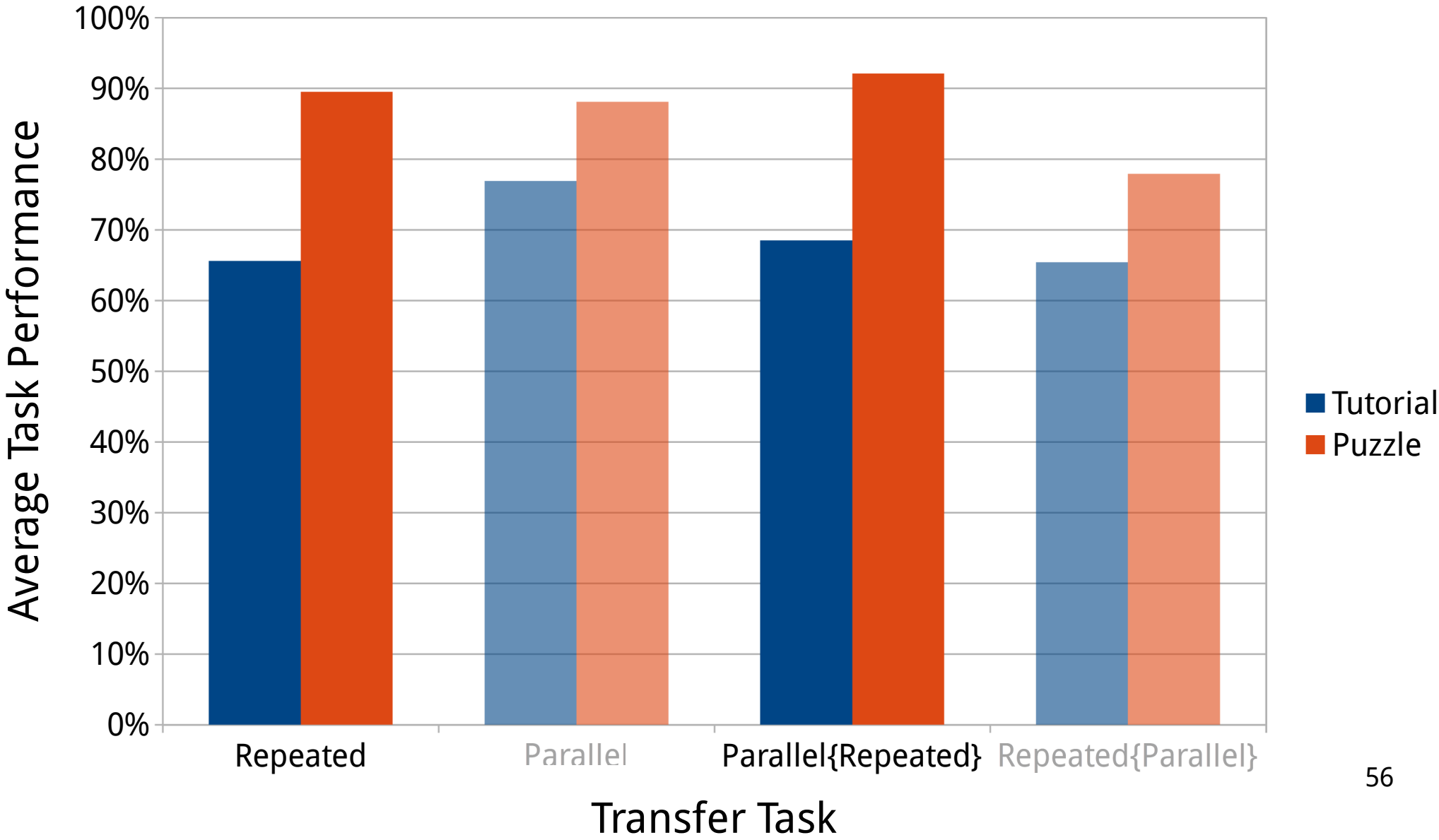
Average Transfer Task Time

$p = .06$

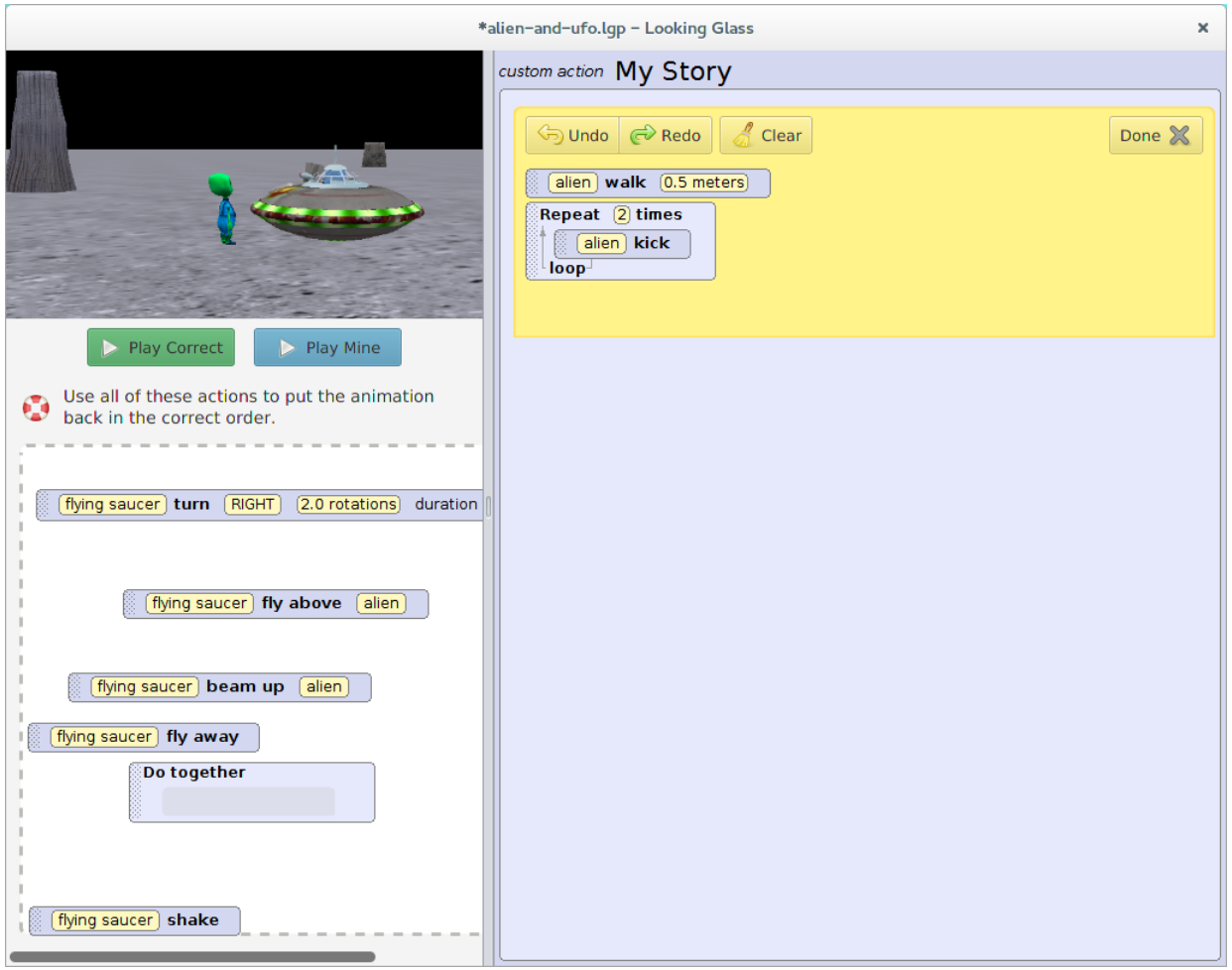


Average Transfer Task Performance

$p < .05$



Summary



Puzzle users performed 26% better on transfer tasks while requiring 23% less training time.

Future Work

- Completion Problems
  - Paired with worked examples
- Distractors
  - Common in puzzle-like systems
  - Impact on completion problem effect

Questions

Kyle J. Harms
Washington University in St. Louis
kyle.harms@wustl.edu

LookingGlass

https://lookingglass.wustl.edu

Washington University in St. Louis