Kodu in Tekkotsu

15-494 Cognitive Robotics
David S. Touretzky

Carnegie Mellon
Spring 2015

Overview

- Implements a subset of Microsoft's Kodu Game Lab.
- No GUI. Kodu source code is read from a text file.
- Features:
  - Perception: “see” and “bump”, “close” and “far”
  - Objects: apple (red), tree (green), rock (blue)
  - Navigation: “move”, “turn”, NSEW and various directions
  - Manipulation: “grab”, “got”, and “drop”
  - Speech and sound effects output (“say” and “play”)
  - Timers, Scores, Randomness
  - Rule dependency (indentation)
  - Multiple pages (mechanism for state machines)

Sample Program

PAGE 1

WHEN see red apple DO move toward

WHEN bump red apple DO grab it

: WHEN DO say “Yum”

: WHEN DO switch_to_page 2

PAGE 2

WHEN see green tree DO move towards

WHEN bump green tree DO drop

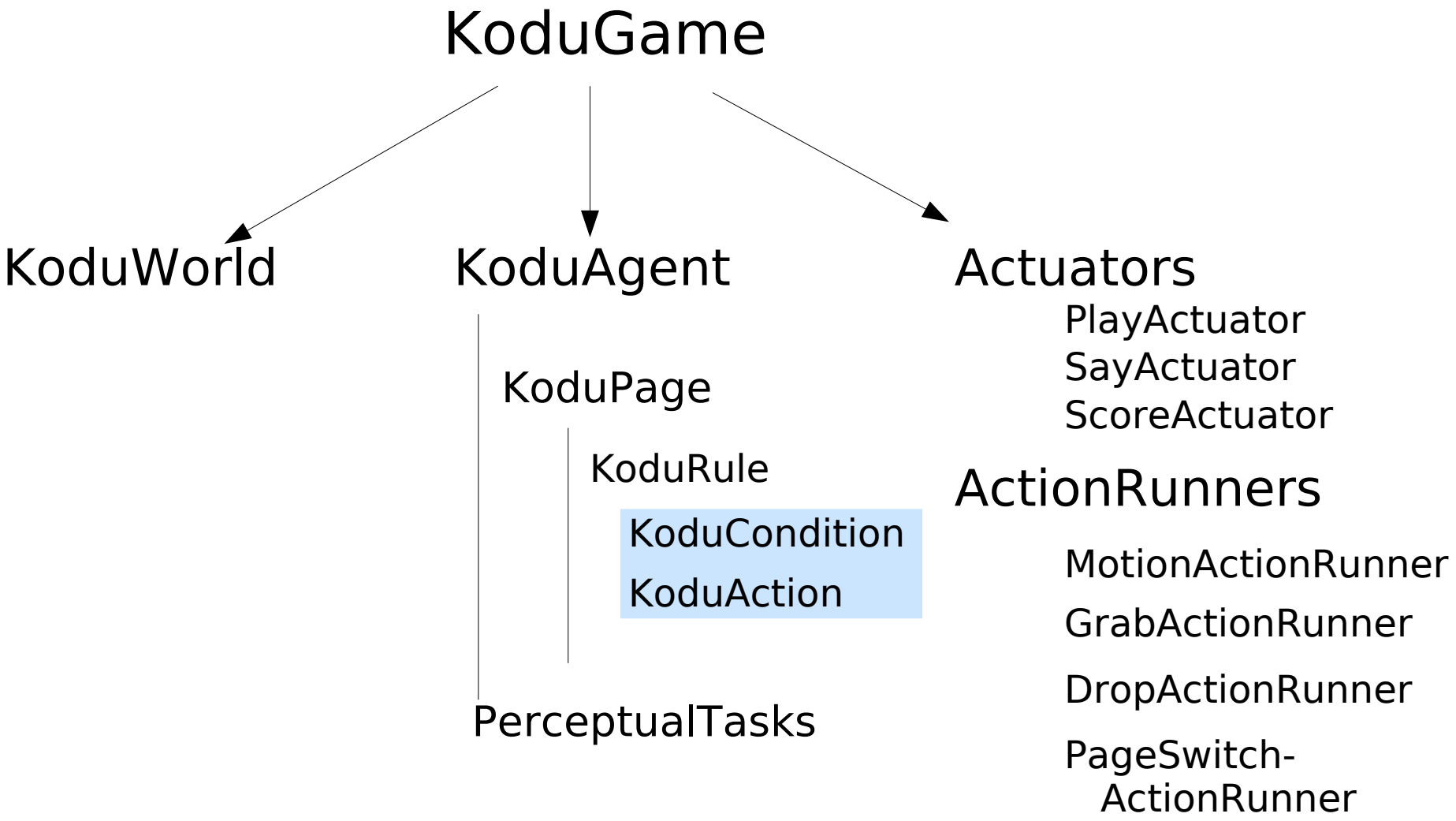
Parsing

- All parsing functions are in the Parsing/ subdirectory.
- A page has a number and an ordered list of rules.
- Rule = WHEN *condition-phrase* DO *action-phrase*
- A phrase has a head token (verb) and optional modifier tokens (nouns, adjectives, adverbs).
- Token types: keywords, numbers, and strings
- Two phrase types: condition phrase, action phrase
- Every head token has a parsing function to verify that the modifiers supplied are valid for that head.

Rule Execution

- All the rules on the current page run in parallel.
- First, all conditions are evaluated.
  - Objects are bound if predicates satisfied, e.g., "red apple" binds to the nearest red apple.
- Next, the actions of rules with true predicates are queued for execution.
  - In case of conflict, the lower numbered rule takes priority.
  - "Switch to page" short circuits any following actions.
- Action runners are separate processes that run independent of the rule interpreter.

Execution Structure



Actuators

- PlayActuator
  - Queues sound files to play
  - Can play multiple sounds at once, but an individual rule can only queue one sound at a time.
- SayActuator
  - The kodu can only say one thing at a time.
  - Maintains a queue of things to be said.
- ScoreActuator
  - Any number of score actions can execute simultaneously.
  - Execution happens in rule order: this is important for non-commutative operations such as “set score”.

ActionRunners

- Physical actions are complex and extended in time, occupying many rule interpreter frames.
- ActionRunners run asynchronously, so the kodu can move, speak, and play sound effects at the same time.
- Some actions require suspension of the rule interpreter until the action completes.
  - Can't perform a “move” during a “grab” or “drop”.
  - Can't switch pages during a “grab” or “drop”.

MotionActionRunner

- Plans a path toward an object, and executes it.
- Can also execute simple motions such as “move north” or “turn left”.
- May stop periodically to do localization.

GrabActionRunner

- Uses the Grasper to pick up an object.
- Does its own failure detection and recovery because the Grasper doesn't do this yet.

DropActionRunner

- Uses the Grasper to drop an object at the current location.

PageSwitchActionRunner

- Page switching must be suspended until the current grab or drop operation has completed.

Perceptual Tasks

- In the PerceptualTasks/ subdirectory.
- Gripper monitoring (for dropped objects).
  - Currently visual, but could use force feedback.
- Visual bump detection.
- Visual localization (AprilTags on walls serve as visual landmarks.)
- Navigation error monitoring: when navigating towards an object, if it's not where you think it should be, then you're not where you think you should be.

Other Code Subdirectories

- Objects
  - Represents Kodu objects, such as apples and trees.
- Generators
  - Generates string or numeric values on demand. These will either be constants, or randomly drawn from a set of allowable values if the “random” tile is used.
- Keepers
  - ScoreKeeper maintains a score.
  - ObjectKeeper maintains a reference to an object.