Uhura: An English -> RMPL/PDDL Translator

1. System requirements
    a. Make sure that you are using a computer that runs one of the following operating systems:
        i. Windows
        ii. Ubuntu
        iii. MAC OS

    b. Make sure that Python 2.7 (or higher) has been installed on your computer. You may download the Python installation file from the following link:

    http://www.python.org/getit/

    c. Install the Natural Language Toolkit (NLTK 2.0). You may find the file and installation instruction on this website:
    http://nltk.org/install.html

    d. We suggest you make a slight modification to the NLTK package in order to have the parse tree displayed correctly. Please open the nltk.draw.util.py file, go to line 1668 and change the original definition:

    ```
    self._canvas = canvas = Canvas(frame, **kw)
    ```

    to

    ```
    self._canvas = canvas = Canvas(parent, **kw)
    ```

    The reason is that once a tkinter widget is created, its parent cannot be changed any more.

    e. (Optional) Install Eclipse and PyDev as the IDE for this package. You may find the installation file and instruction on the following two websites:

    http://www.eclipse.org/
    http://pydev.org/download.html

    Please remember to set Python2.7 as the interpreter for Eclipse after you installed PyDev. We will be using Eclipse throughout this instruction file to demonstrate how to use Uhura. However, it is ok if you just launch the 'LaunchTkinterGUI.py' directly using a shell command or other IDEs.

    f. Download the Uhura package through the following link:
    http://people.csail.mit.edu/yupeng/files/UhuraPython.zip

2. Import the project
    a. Open Eclipse.

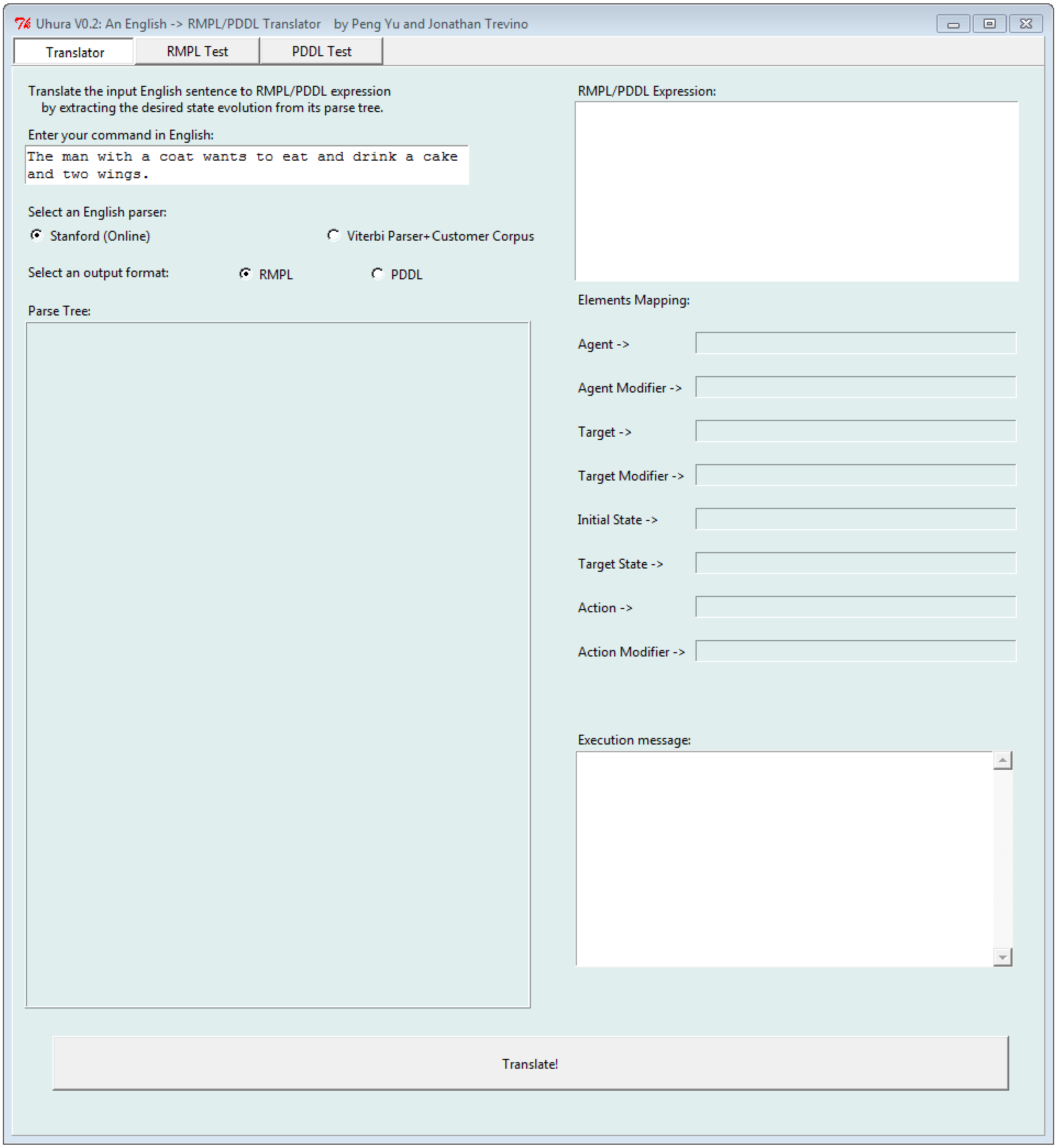
    b. Unzip UhuraPython.zip to a folder.

c. Import the UhuraPython project:
   Choose File ->
   Choose Import ->
   select ‘General’ then ‘Existing Projects into Workspace’ ->
   click Next ->
   Enter the direction of the folder of ‘UhuraPython’ ->
   click Finish.

d. Make sure that the interpreter for this project is Python2.7:
   Right click ‘UhuraPython’ in the Package Explorer ->
   Select PyDev – Interpreter/Grammar ->
   Make sure that the Grammar Version is 2.7, and the Interpreter is Python2.7 (not default).

e. Launch Uhura:
   Find the file ‘launchTkinterGUI.py’ ->
   Right click ->
   Select ‘Run As’ ->
   Click ‘Python Run’

f. You will see the Uhura GUI being launched, like the following picture:


Uhura V0.2: An English -> RMPL/PDDL Translator by Peng Yu and Jonathan Trevino

Translator | RMPL Test | PDDL Test

Translate the input English sentence to RMPL/PDDL expression
by extracting the desired state evolution from its parse tree.

Enter your command in English:

```
The man with a coat wants to eat and drink a cake
and two wings.
```

Select an English parser: [✓ Stanford (Online)] Viterbi Parser+Customer Corpus

Select an output format: [✓ RMPL] PDDL

Parse Tree:

RMPL/PDDL Expression:

Elements Mapping:

Agent ->

Agent Modifier ->

Target ->

Target Modifier ->

Initial State ->

Target State ->

Action ->

Action Modifier ->

Execution message:

3. Translate English sentences using Uhura
    a. Parser selection

Uhura comes with two different English parsers: Stanford and a customized Viterbi parser. The Stanford parser is developed by the Natural Language Processing group at Stanford University. Uhura accesses their online parser (http://nlp.stanford.edu:8080/parser/) through a HTML query and extracts the parse tree from the returned webpage. Therefore, you need Internet connection in order to use this parser.

Pros: It parses almost everything and runs very fast (since we do not compute anything locally).
Cons: The quality of the parsing is not quite accurate, especially for the HIGH/LOW attachments of propositional phrases. We do not have the access to the corpus to improve the quality.

The second one is a probabilistic parser that is trained using the Wall Street Journal corpus. It generates the most like parser of a given English sentence using the Viterbi algorithm.

Pros: It is customized so that we can add new rules to it and further customize it for parsing English commands. Runs locally so that you do not need Internet connection to use it.
Cons: It is very slow…and the vocabulary is very limited. You may encounter the '`ValueError: Grammar does not cover some of the input words`' very often.

Select one parser by clicking on the corresponding radio button.

b. Output selection

Uhura can translate English sentences to either RMPL or PDDL expressions.

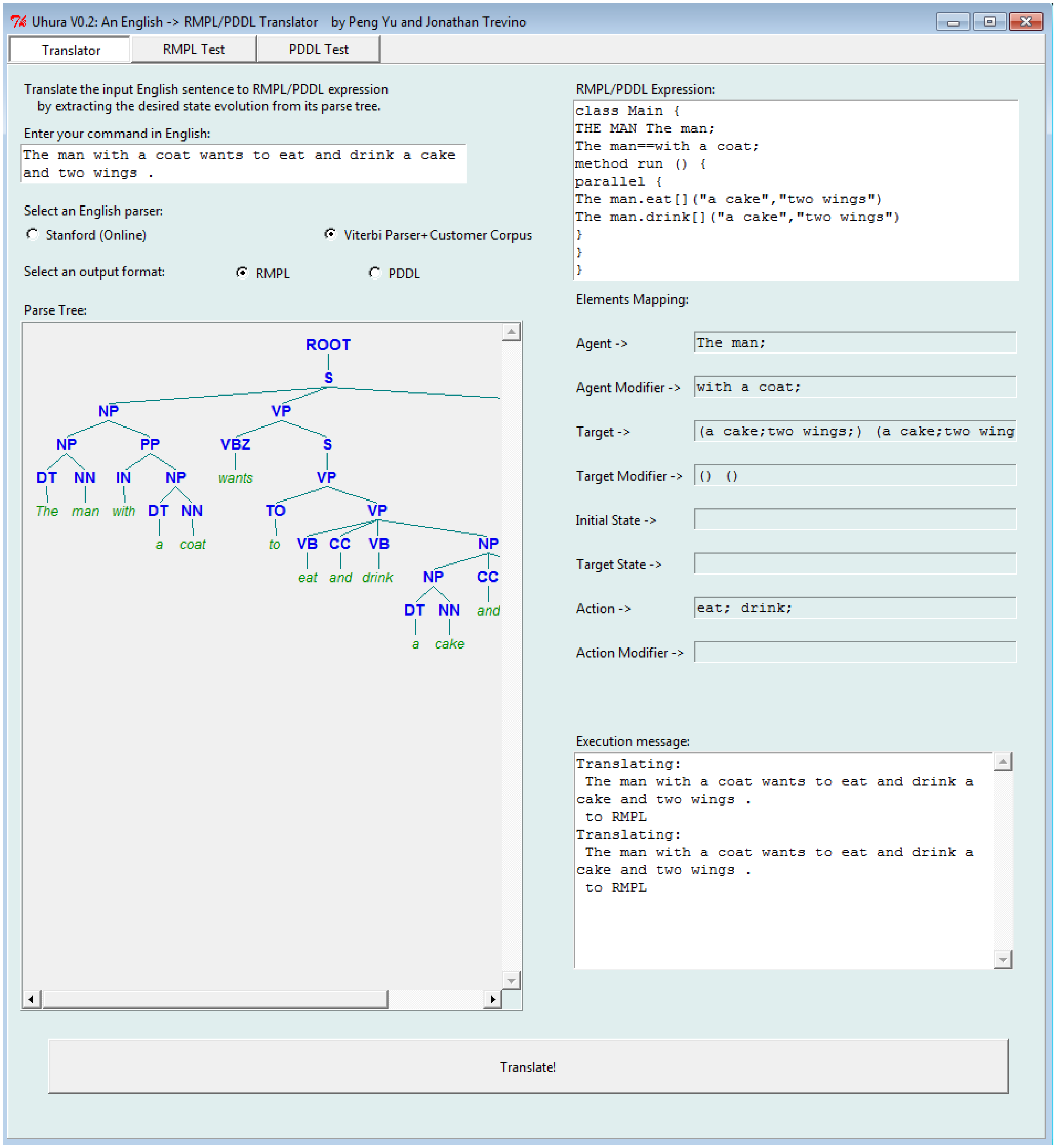
i. RMPL
Reactive Model-based Programming Language (RMPL) is a programming language developed by the MIT MERS group. It provides a framework for constraint-based modeling and reactive programming constructs. It is especially suitable for capturing the users' goals in a planning problem. The reference can be found in the paper *A Reactive Model-based Programming Language for Robotic Space Explorers* (http://groups.csail.mit.edu/mers/papers/isairas01_rmpl.pdf).

ii. PDDL
Planning Domain Definition Language (PDDL) was developed to standardize the planning language in artificial intelligence research. It is extremely useful in the description of planning domains as well as the states of the world. More information can be found at (http://ipc.icaps-conference.org/).

Select one output by clicking on the corresponding radio button.

c. Translate an English sentence

Simply type a sentence into the text box under label 'Enter your command in English', and then click the button 'Translate!' located at the bottom of the page. The results will be displayed in the right side of the GUI window, like the following picture.



Translate the input English sentence to RMPL/PDDL expression by extracting the desired state evolution from its parse tree.

Enter your command in English:

```
The man with a coat wants to eat and drink a cake
and two wings .
```

Select an English parser: Stanford (Online) [✓ Viterbi Parser+Customer Corpus]

Select an output format: [✓ RMPL] PDDL

Parse Tree:

RMPL/PDDL Expression:

```
class Main {
THE MAN The man;
The man==with a coat;
method run () {
parallel {
The man.eat[]("a cake","two wings")
The man.drink[]("a cake","two wings")
}
}
}
```

Elements Mapping:

Agent ->	The man;
Agent Modifier ->	with a coat;
Target ->	(a cake;two wings;) (a cake;two wing
Target Modifier ->	() ()
Initial State ->	
Target State ->	
Action ->	eat; drink;
Action Modifier ->	

Execution message:

```
Translating:
 The man with a coat wants to eat and drink a
cake and two wings .
 to RMPL
Translating:
 The man with a coat wants to eat and drink a
cake and two wings .
 to RMPL
```

The parse tree returned by the parser is displayed in the Parse tree window. Uhura tries to extract eight types of information from the parse tree, namely:

- Agent
- Agent modifier
- Target

- Target modifier
- Initial state
- Target state
- Action
- Action modifier

Uhura then generates the corresponding RMPL or PDDL expressions using these data. The result is shown in the top right corner of the GUI window.

4. Interacting with Activity Planner using Uhura

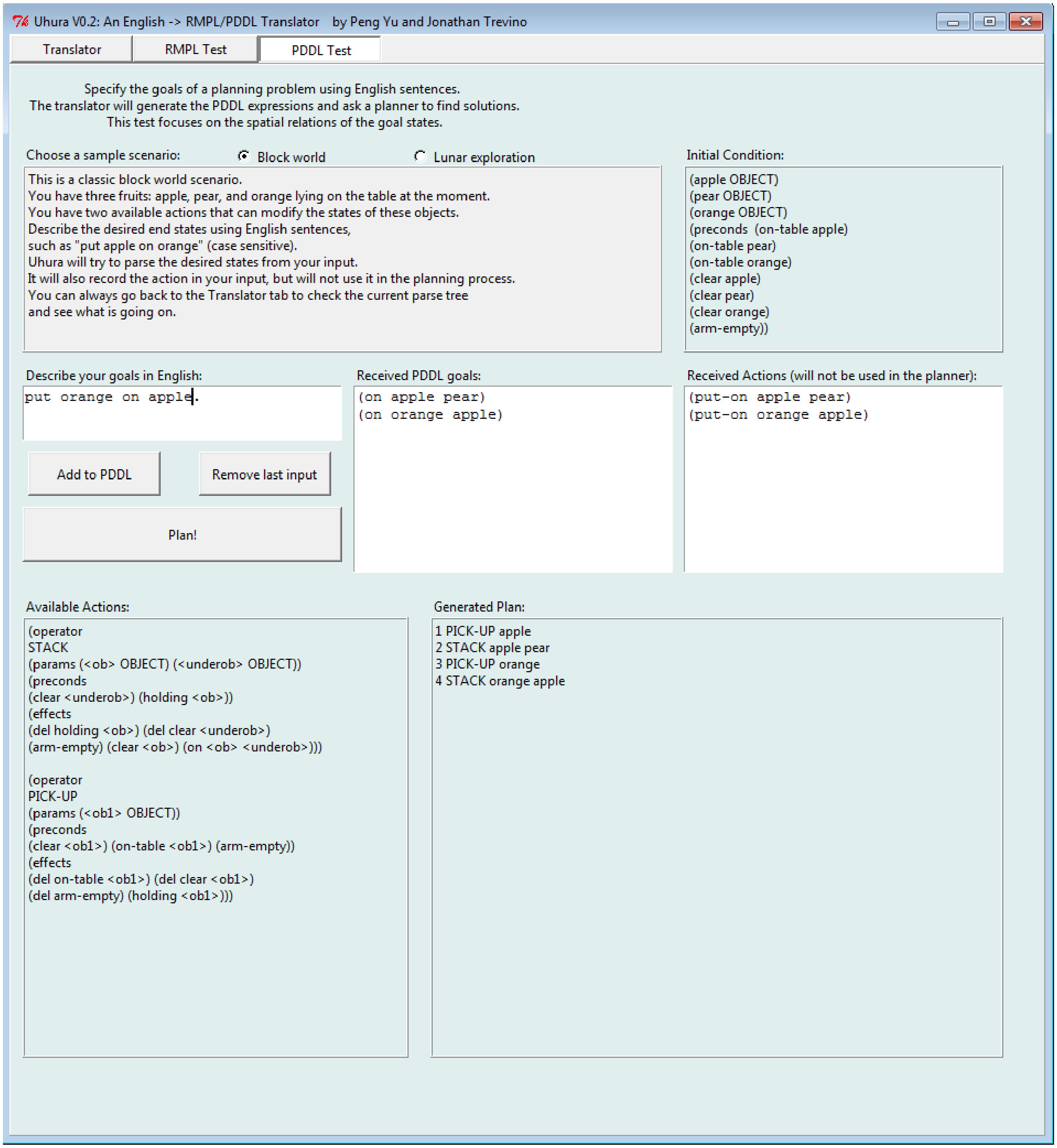
Uhura is designed to serve as a natural language interface between the user and planning algorithms. Here we include two tests that demonstrate this capability.

a. RMPL
Unfortunately, our RMPL planner ‘Kirk’ (*P. Kim, B.C. Williams, M. Abramson, Execution of reactive model-based programs through graph-based temporal planning, in: Proceedings of the 17th International Joint Conference on AI (IJCAI-01), Seattle, WA, 2001, pp. 487–493*) is not ready to support the planning task in unspecified domains. This tab now serves as a RMPL program generator. You may describe your objectives towards the sample scenario into a RMPL program. We will provide the planner later in an update.

b. PDDL
Uhura comes with the GraphPlan software, which is a general-purpose planner for STRIPS-style problems (We will move to FF planner in the future, which is capable of processing PDDL problems and runs faster).

To interact with the planner, simply input your description of the goals states using English. Uhura will extract the goal state description from your inputs. However, it will not preserve your input of actions. For example, if you input ‘I want to place orange on apple.’ Uhura only takes the ‘orange on apple’ part and ignores your action description. We are working on resolving this problem by enabling the user to define new action.

There are two problems you may work with: one is in the classic block world scenario; the other one is in the lunar surface transportation scenario. Choose the scenario you want to work in, add your descriptions of desired goal states by clicking ‘Add to PDDL’, remove any unwanted goals by clocking ‘Remove last input’ and call the planner for a solution by clicking the ‘Plan!’ button. If a solution exists, it will be displayed in the bottom right corner of the window as shown in the following image. If not, Uhura will notify you that no feasible plan can be found.


Uhura V0.2: An English -> RMPL/PDDL Translator by Peng Yu and Jonathan Trevino
Translator
RMPL Test
PDDL Test

Specify the goals of a planning problem using English sentences.
The translator will generate the PDDL expressions and ask a planner to find solutions.
This test focuses on the spatial relations of the goal states.

Choose a sample scenario: [✓ Block world] Lunar exploration

This is a classic block world scenario.
You have three fruits: apple, pear, and orange lying on the table at the moment.
You have two available actions that can modify the states of these objects.
Describe the desired end states using English sentences,
such as "put apple on orange" (case sensitive).
Uhura will try to parse the desired states from your input.
It will also record the action in your input, but will not use it in the planning process.
You can always go back to the Translator tab to check the current parse tree
and see what is going on.

Initial Condition:

```
(apple OBJECT)
(pear OBJECT)
(orange OBJECT)
(preconds  (on-table apple)
(on-table pear)
(on-table orange)
(clear apple)
(clear pear)
(clear orange)
(arm-empty))
```

Describe your goals in English:

```
put orange on apple.
```

Received PDDL goals:

```
(on apple pear)
(on orange apple)
```

Received Actions (will not be used in the planner):

```
(put-on apple pear)
(put-on orange apple)
```

Available Actions:

```
(operator
STACK
(params (<ob> OBJECT) (<underob> OBJECT))
(preconds
(clear <underob>) (holding <ob>))
(effects
(del holding <ob>) (del clear <underob>)
(arm-empty) (clear <ob>) (on <ob> <underob>)))

(operator
PICK-UP
(params (<ob1> OBJECT))
(preconds
(clear <ob1>) (on-table <ob1>) (arm-empty))
(effects
(del on-table <ob1>) (del clear <ob1>)
(del arm-empty) (holding <ob1>)))
```

Generated Plan:

```
1 PICK-UP apple
2 STACK apple pear
3 PICK-UP orange
4 STACK orange apple
```

5. About Uhura
   Uhura is written by Peng Yu and Jonathan Trevino as our final project for the course 6.863 Natural Language Processing. We would like to keep improving this software and your comments are greatly appreciated. Please send your comments or any bug/error reports to Peng Yu (yupeng@mit.edu). Thanks!