R2D3 Robotic Development 3nvironment

Version 0.2 Manual (Unfinished)

1 About R2D3

R2D3 is a Free, Cross Platform, Generic Robotic Manipulator Simulator and Development Environment built using Blender and Python. It's main goal is to provide a simple but feature-rich platform for Educational & Research Usage. Much of it is currently Scorbot ER-V (From Intelitek) oriented though that situation *will* change in the near future.

2 Blender + Python ROCKS!

Blender and Python make an amazing combo! When I started this project I didn't understand either Blender or Python (but they both looked cool, and that opinion has only strengthened with familiarity). Without either of them this project would not exist.

3 Release 0.2

The code is far from pretty and there is a lot of scope for improvement. Also this is ALPHA Software so not all functionality is implemented yet. Simultaneously, Ver 0.2 is the first 'good-looking' version of R2D3, that's also Semi-useful :) Enjoy!

4 R2D3 Features (Not all implemented)

4.1 Plugin based system (Not yet implemented)

A plugin based system that takes in pluggable robot description files and pluggable scene description files and can thus simulate the operations of most types of robotic manipulator arms.

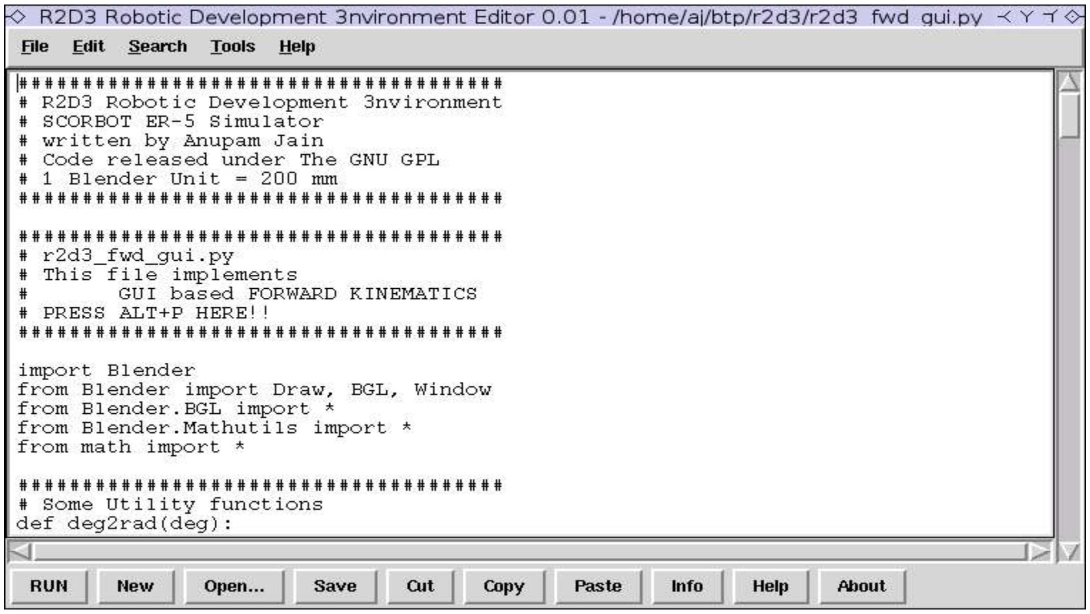
4.2 Real time 3D view

Courtesy Blender :)

4.3 Batch and Remote Control (Not yet implemented)

The software should allow control over the robots through various means i.e. through batch scripts that allow the robot to be run without assistance from the programmer. Also the system, which acts as an interface between the human and the robot, should allow the robot to be controlled over a network provided the software is installed on both the client and the server sides.

4.4 Integrated Program Code Editor (Partially Implemented)


R2D3 Robotic Development 3nvironment Editor 0.01 - /home/aj/btp/r2d3/r2d3_fwd_gui.py
File Edit Search Tools Help
RUN New Open... Save Cut Copy Paste Info Help About

```
######################################
# R2D3 Robotic Development 3nvironment
# SCORBOT ER-5 Simulator
# written by Anupam Jain
# Code released under The GNU GPL
# 1 Blender Unit = 200 mm
######################################

######################################
# r2d3_fwd_gui.py
# This file implements
#       GUI based FORWARD KINEMATICS
# PRESS ALT+P HERE!!
######################################

import Blender
from Blender import Draw, BGL, Window
from Blender.BGL import *
from Blender.Mathutils import *
from math import *

######################################
# Some Utility functions
def deg2rad(deg):
```

Based on Editor.py

4.4.1 Programming Using Real World Physical Units (Partially Implemented)

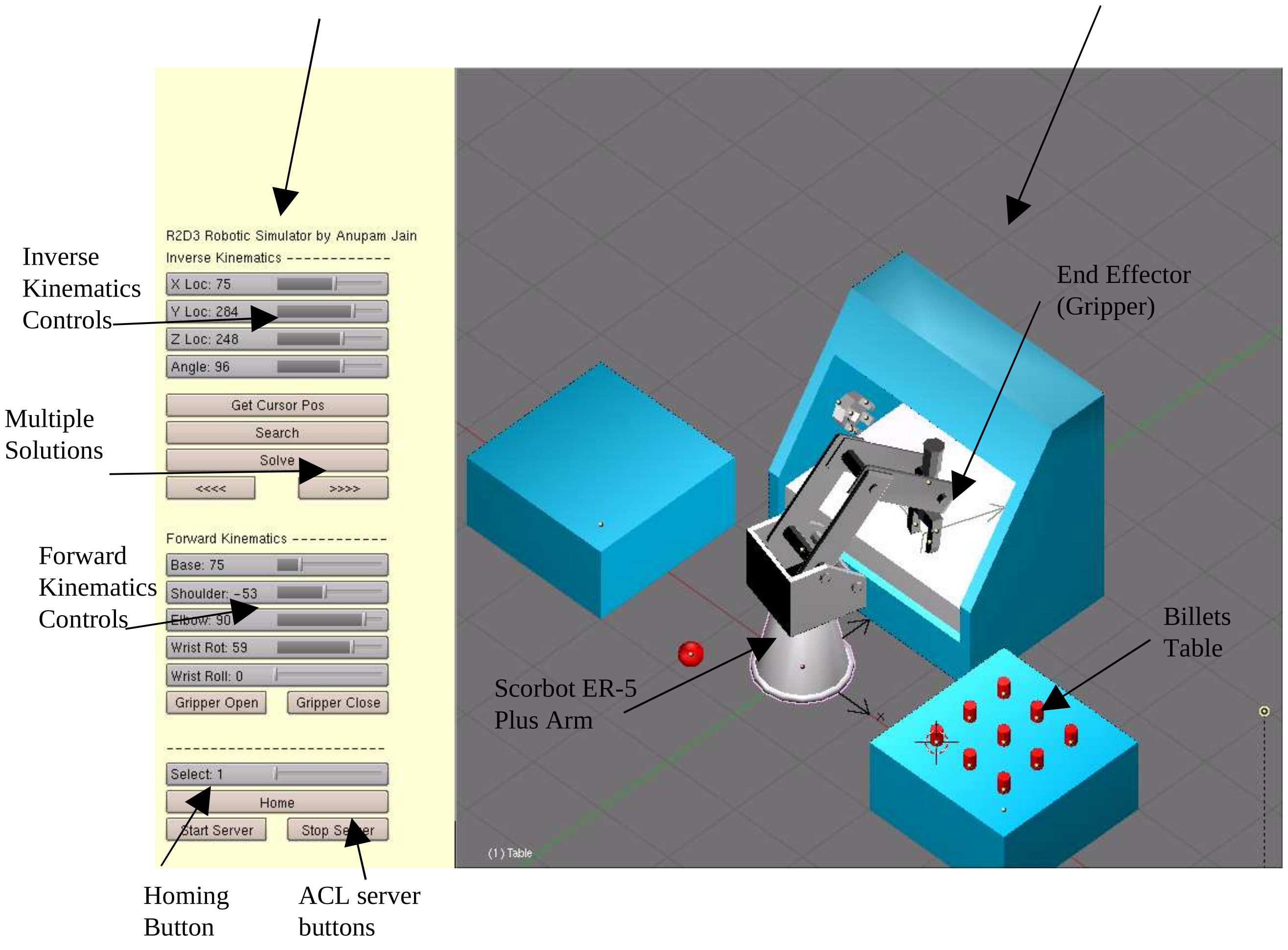
The simulator uses a ratio of 200 for the real world to Blender units translation. i.e. -

1 Blender Unit = 200 mm

4.5 The Simulator Operation

The simulator screen is divided into two separate parts -

The Graphical User Interface Controls **The 3D workspace**



4.5.1 The Graphical User Interface Controls

The graphical user controls are divided into three sections -

4.5.1.1 Forward Kinematics Controls

The forward kinematics controls allow control over the Robot Joint Angles. There are Five separate sliders that provide control over the five separate joints of the 5 DOF Scorbot ER – 5 Plus Robot.

In all the angles, only the angles within range of the actual hardware are allowed. The allowable angle ranges are as follows -

1. Base :	+0 to +310 degrees
2. Shoulder :	-130 to +35 degrees
3. Elbow :	-130 to +130 degrees
4. Wrist Rotation :	-130 to +130 degrees
5. Wrist Roll :	+0 to +570 degrees

In addition there are two buttons to control the gripper -

1. Gripper Open
2. Gripper Close

4.5.1.2 Inverse Kinematics Controls

The inverse Kinematics Controls consist of Four different slider for controlling the following parameters -

1. 'X' Location of the gripper
2. 'Y' Location of the gripper
3. 'Z' Location of the gripper
4. The 'Orientation' of the gripper in the plane of approach of the target.
   The orientation is equal to the sum of the Shoulder Elbow and Wrist Rotations.
   It denotes the angle with which the gripper approaches the target.

All these four parameters can be specified by moving the slider or typing a value in the text box. They can be manipulated independently for any given target orientation of the gripper in order to perform inverse kinematics. Also the modification of any of these does not affect the “Wrist Roll' of the manipulator. Thus the 'Wrist Roll' slider can also be manipulated to perform inverse kinematics.

The combination of these five values fully and unambiguously specifies the end effector orientation of the target position.

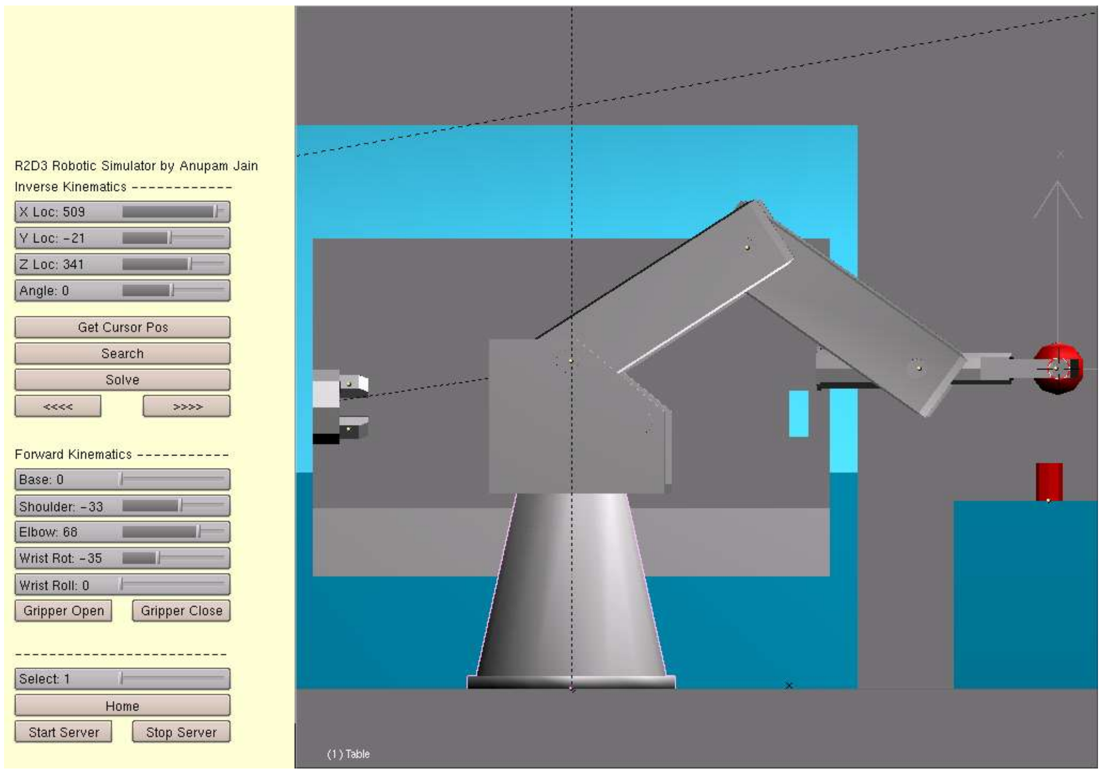
In addition, there are five separate buttons -

1. 'Get Cursor Position' lets the user specify the target location by clicking on the 3d workspace instead of having to manually enter values. The point chosen is always parallel to the current viewing plane.
2. 'Search' performs a searching routine in order to reach the target point. Note that it does NOT perform any analytical calculations and the searching is purely numerical step search.
3. 'Solve' uses the inverse kinematics equations used in the inverse kinematics routine in order to get the corresponding solution. It then moves the simulated robotic arm to that orientation.
4. The solve routine for the inverse Kinematics gives multiple solutions. Thus two 'Forward' and 'Backward' Buttons are provided to cycle between the multiple solutions 'If any'.

The two typical configurations for a robot are -

1. *Elbow Up*
2. *Elbow Down*

Both can be clearly seen in the following screen shots.


R2D3 Robotic Simulator by Anupam Jain

Inverse Kinematics ------------

X Loc: 509

Y Loc: -21

Z Loc: 341

Angle: 0

Forward Kinematics -----------

Base: 0

Shoulder: -33

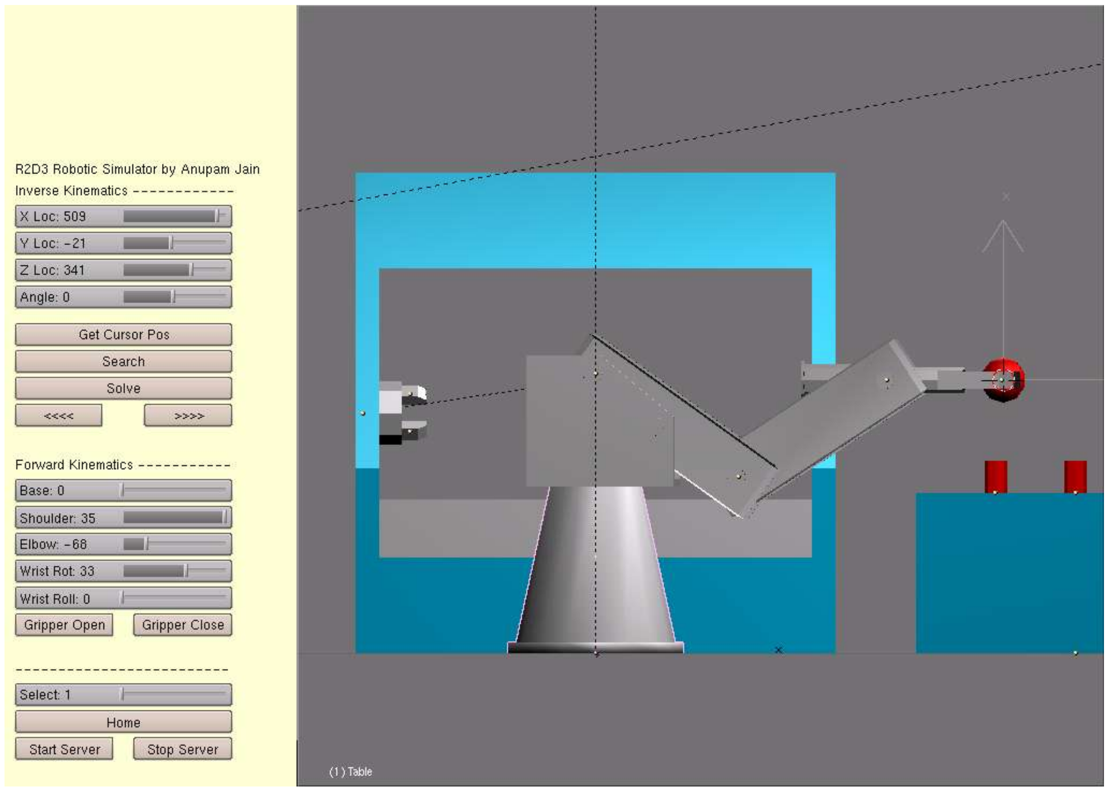
Elbow: 68

Wrist Rot: -35

Wrist Roll: 0

Select: 1

Elbow-Up configuration for a target position


R2D3 Robotic Simulator by Anupam Jain

Inverse Kinematics ------------

X Loc: 509

Y Loc: -21

Z Loc: 341

Angle: 0

Forward Kinematics -----------

Base: 0

Shoulder: 35

Elbow: -68

Wrist Rot: 33

Wrist Roll: 0

Select: 1

Elbow-Down configuration for the same target position

4.5.1.3 Miscellaneous Controls

The miscellaneous controls have buttons for the other features provided by the project.

1. 'Homing' Routine
   A button is provided to send the robot to its home position.

2. 'Select' Billet Slider
   The slider can be used to select the billets provided on the workpiece table (Numbered from Zero to Nine). The billet selected automatically has its position coordinates fed into the inverse kinematic sliders so that the user just has to press the 'Solve' Button to pick up the billet.

3. 'Start Server'
   Starts the ACL Server inbuilt into the software. It listens to the specified port using TCP/IP protocol and can accept ACL commands to control the simulator. Any external program whether on the same computer or not, using the network interface, can control the simulator. The program can accept at most five simultaneous connections from clients.

   The ACL commands can also be sent to the real Robot over the serial line.

4. 'Stop Server'
   Stops the ACL server.

5. 'Editor'
   Starts the Inbuilt Code Editor. It can be used to store upto ten ACL programs that can be executed at the click of a button. The editor provides the usual code editing facilities like searching, line count etc. Upon clicking 'Run', the editor runs the program in realtime. It can also control the actual robot over the serial link.