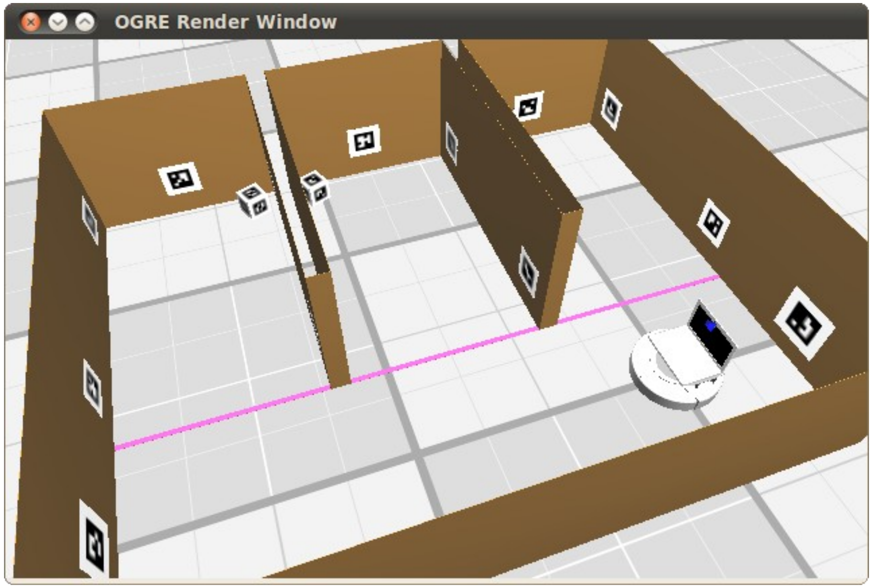
The Mirage Simulator and Virtual Worlds

15-494 Cognitive Robotics
David S. Touretzky &
Ethan Tira-Thompson

Carnegie Mellon
Spring 2012

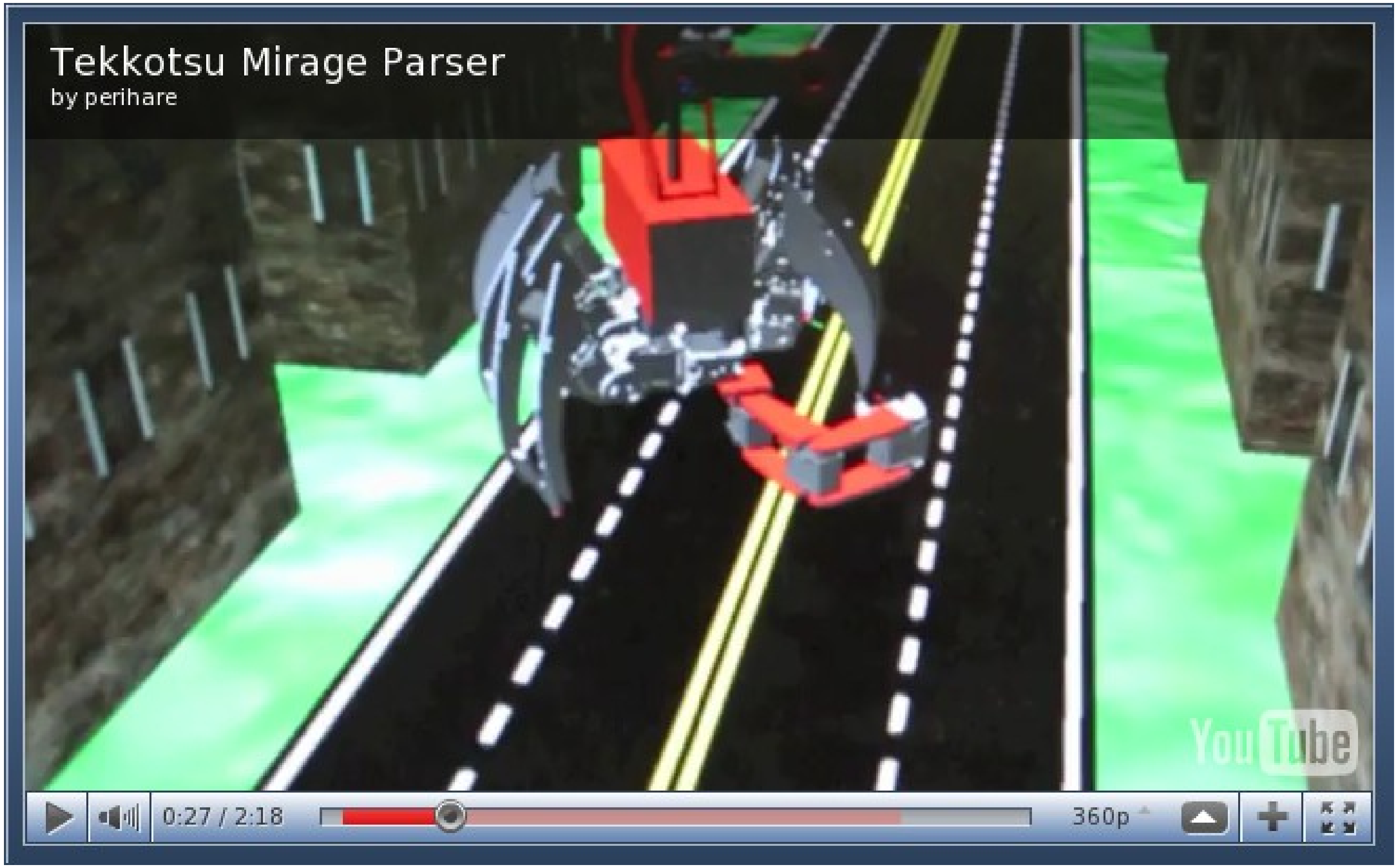
What is Mirage?

- A virtual environment simulator for Tekkotsu.
- Built on top of:
  - OGRE3D rendering engine
  - Bullet physics engine



- Provides Tekkotsu with simulated camera images and sensory inputs (e.g., bump sensors on the Create).
- Provides the user with a 3D rendering of a simulated robot and its environment.
- See the Tekkotsu wiki for documentation.

Mirage Demo Video



http://www.youtube.com/watch?v=JO1gW4BKS50

Pros and Cons of Simulation

Pros:

- Perfect repeatability: allows for controlled experiments.
- Safely simulate conditions that could be dangerous to the robot, e.g., walking on stairs.
- Simulate variations on existing robots, new robots that haven't been built yet, or robots you can't afford to buy.

Cons:

- Physics is not accurate.
- Lighting is not realistic.
- Simulator may not handle all functions, e.g., buttons on the Create, depth mapping on Calliope5KP.

Running Mirage

Make three tabs in your terminal window:

- Third tab:

```
> ControllerGUI localhost
```

- Second tab:

```
> cd /usr/local/Tekkotsu/tools/mirage
> ./mirage  worlds/tic-tac-toe.mirage
```

- First tab:

```
> cd ~/project
> ./tekkotsu-CREATE  -c mirage.plist
```

Mirage Command Summary

- Viewpoint motion:

  w = forward, s = back

  a = move left, d = move right

  PageUp = climb, PageDown = descend

  Arrow keys = pitch up/down and yaw left/right

- Other basic commands:

  r = reload the world

  u = unfollow / view reset

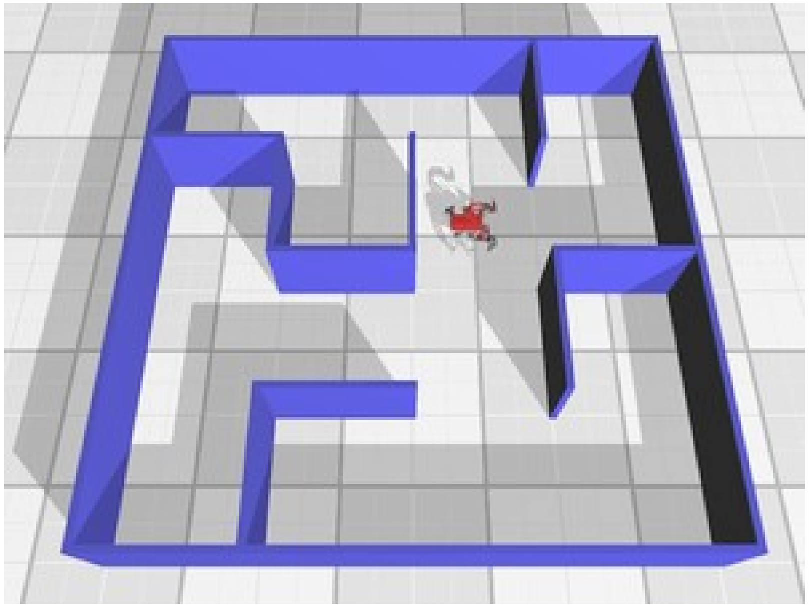
  t = print statistics

  q = quit

- See the Tekkotsu wiki for additional commands.

Mirage Worlds

- World files with extension .mirage are xml files: human readable but ugly.
- Specify world file on the command line and it will be loaded by Mirage during startup.
- Tekkotsu includes a collection of predefined worlds in
  /usr/local/Tekkotsu/tools/mirage/worlds



Mirage World Builder Tool

- Provides a convenient, simple syntax for constructing Mirage worlds.
- Source files have extension .ian.
- WorldBuilder tool translates these into .mirage files.
- Most Mirage worlds are built using this tool; the source for these worlds can be found in:

  /usr/local/Tekkotsu/tools/mirage/worlds/src

- Users can put their own world files in
  ~/project/worlds

WorldBuilder Language

- Shape commands: create shapes in the world
  - Simple shapes: cube, sphere, cylinder, plane
  - Models: complex shapes defined in mesh files created using a 3D modeling tool such as Blender
- Light sources
  - Define position, direction, and color of each light source
- Shadows: rendering parameters
- Physics: gravity, space & mass scale, granularity

WorldBuilder Shape Attributes

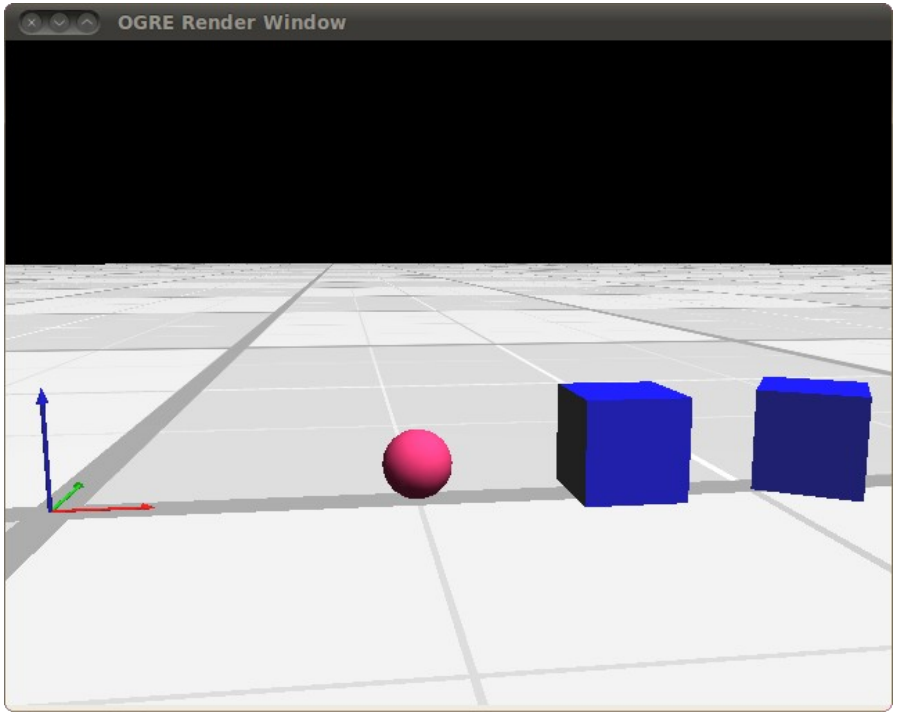
sphere scale=[50,50,50] location=[250,0,25] material=Pink

- Location specifies the *center* of the object, not the bottom!
- Basic materials such as Pink are defined in /usr/local/Tekkotsu/tools/mirage/media/General.material

cube scale=[75,75,75] location=[400, 0, 75/2] material=Blue

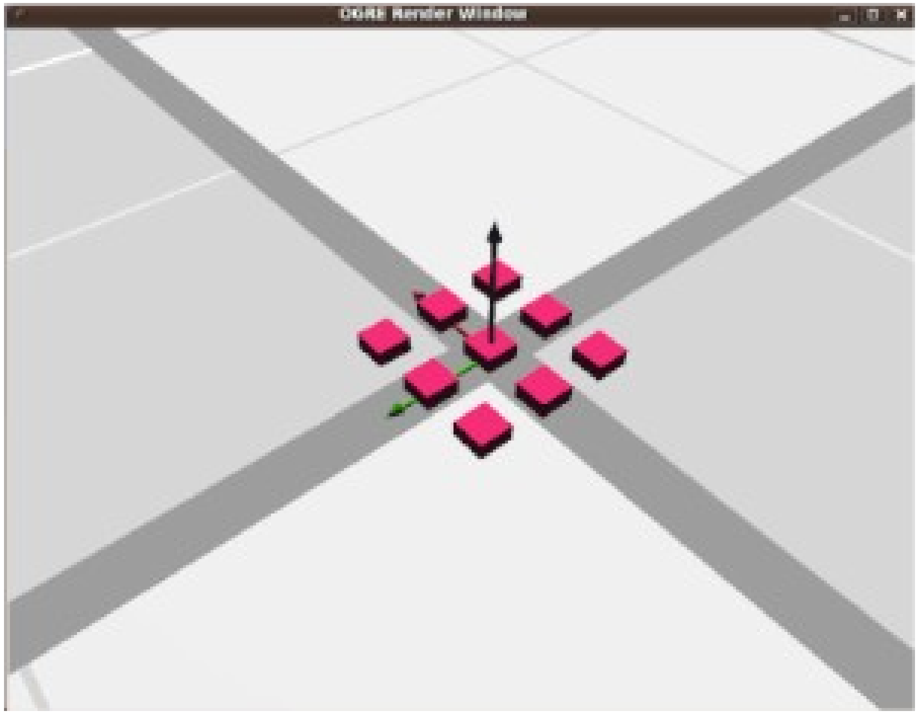
cube scale=[75,75,75]
location=[550, 0, 75/2]
material=Blue orientation=rotz(1)

- Rotation about the z axis by 1 radian



Defining New Shapes

```
define pinkcube cube scale=[20,20,20] material=Pink

pinkcube location=[0,0,0]
pinkcube location=[-40,0,0]
pinkcube location=[40,0,0]

pinkcube location=[0,40,0]
pinkcube location=[-40,40,0]
pinkcube location=[40,40,0]

pinkcube location=[0,-40,0]
pinkcube location=[-40,-40,0]
pinkcube location=[40,-40,0]
```



Why are these cubes embedded in the floor?

Expressions

- Attributes can contain arithmetic expressions.
  - cube scale=[8*8, 6*sqrt(11), 2+4]

- The **set** command assigns values to variables.

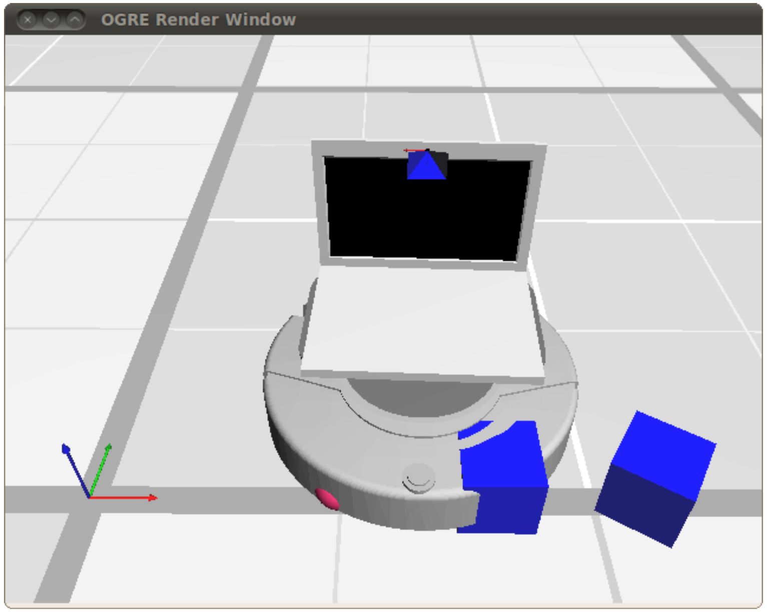
```
set cubesize = 125

set cube_z = cubesize/2

cube scale=[cubesize,cubesize,cubesize]
    location=[0, 0, cube_z]
```

Mass Attribute

- By default, objects have zero mass.
- Objects with no mass are collidable but immovable:
  - sphere scale=[50,50,50] location=[250,0,25] material=Pink
  - Using the Walk Controller, try driving the Create into this zero mass object.
- A modest mass (in kg) makes objects pushable.
  - sphere scale=[50,50,50] location=[250,0,25] material=Pink **mass=5**
- A large mass provides greater resistance.
  - sphere scale=[50,50,50] location=[250,0,25] material=Pink **mass=5000**

Collision Attribute

- Setting collision=false for an object disables collisions.
- Allows the robot to pass right through the object.
- This is useful for making markings in the environment (such as boundary lines) that shouldn't impede the robot's motion.
  - cube scale=[75,75,75] location=[400, 0, 75/2] material=Blue **collision=false**



Running the WorldBuilder

- Put your .ian files in ~/project/worlds
- To generate a .mirage file from myworld.ian, do:

```
> cd ~/project/worlds
> WorldBuilder myworld.ian
```

- To run Mirage using this world, do:

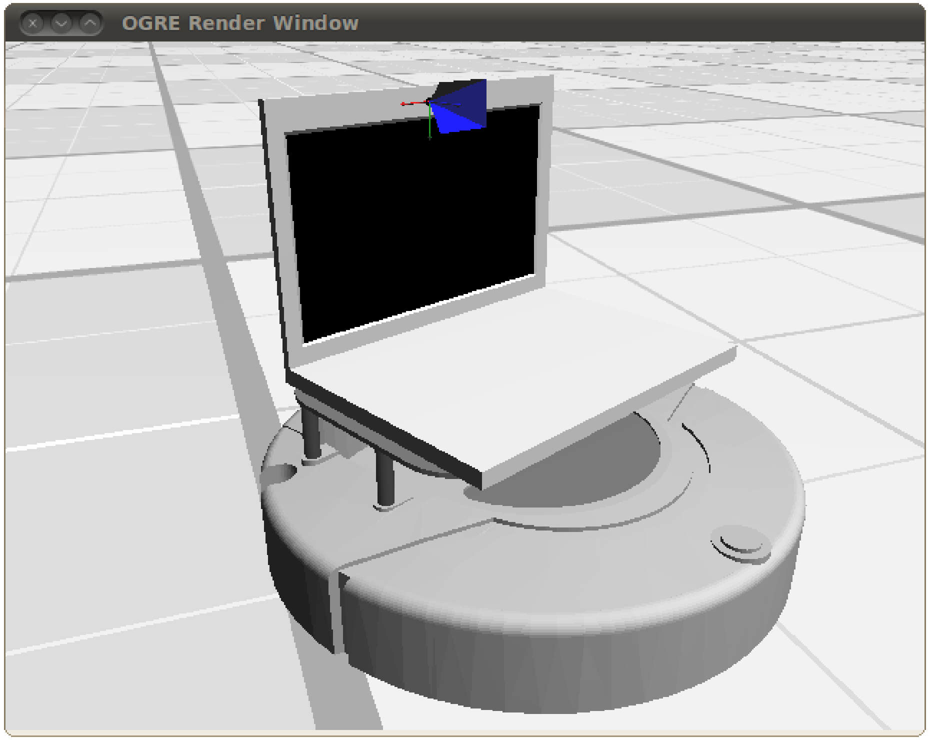
```
> cd /usr/local/Tekkotsu/tools/mirage
> ./mirage ~/project/worlds/myworld.mirage
```

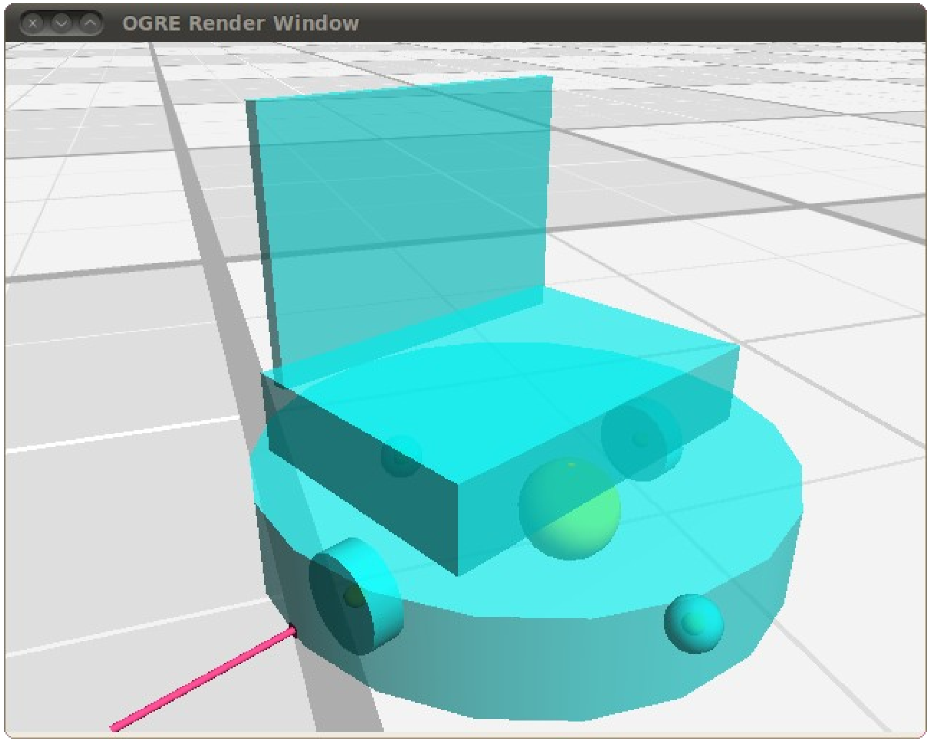
How Is The Robot Defined?

- “kin” file in project/ms/config defines the kinematic structure of the robot.
  - Ex: project/ms/config/Create.kin
  - These kin files are XML files.
- Entries in the kin file refer to mesh files for the various components of the robot, using the **Model** attribute.
- Mesh files can be found in, e.g.,
  /usr/local/Tekkotsu/tools/mirage/media/Create

Collision Models

- Collision detection for complex shapes is expensive.
- Solution: provide a simpler collision model.
- The “h” command toggles collision model display.





Material Files

- Materials are defined in material files. Some basic materials are defined in:
  - /usr/local/Tekkotsu/tools/mirage/media/General.material
- Material files can be found in two places:
  - System materials: /usr/local/Tekkotsu/tools/mirage/media/
  - User-defined materials: ~/project/worlds
- Material files are loaded automatically by Mirage.
- Information on material definition syntax is available in the OGRE3D documentation.

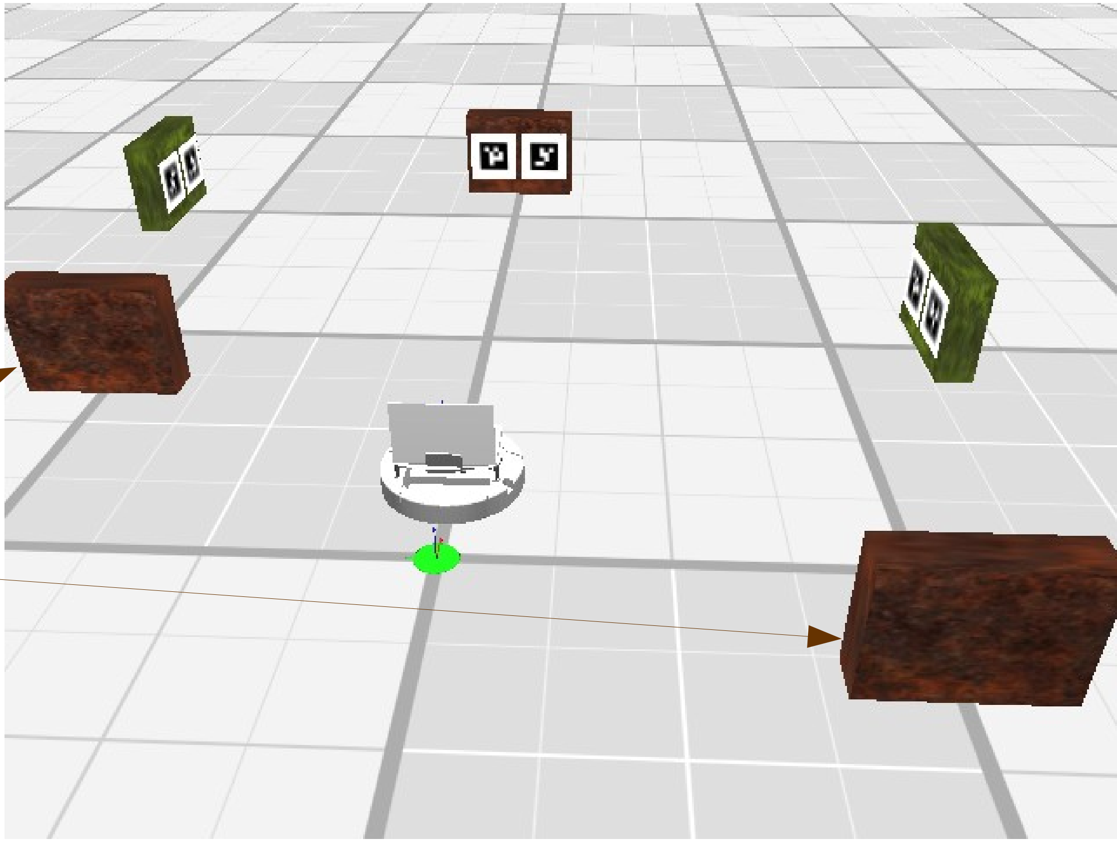
Simple Materials

An example from the General.material file:

```
material Pink
{
    receive_shadows on
    technique
    {
        pass
        {
            ambient 0.4 0.1 0.2 1.0
            diffuse 1 0.25 0.5 1
            emissive 0 0 0 1
        }
    }
}
```

Textured Materials From Images

```
material RustySteel
{
    receive_shadows on
    technique
    {
        pass
        {

            texture_unit
            {
                texture RustySteel.jpg
            }
        }
    }
}
```

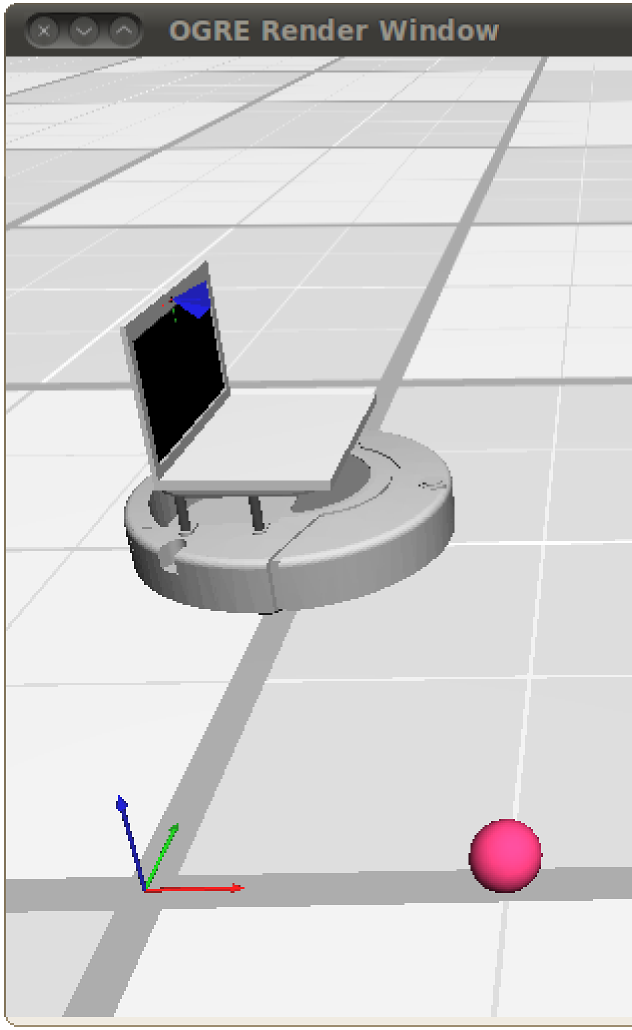


Initializing Robot Pose

- You can specify InitialLocation and InitialOrientation parameters to the Mirage driver when starting Tekkotsu.
- Components 0,1,2 correspond to x,y,z.
- Orientation is a unit quaternion.
- Start robot 500 mm west of the origin:

```
./tekkotsu-CREATE -c mirage.plist
    Drivers.Mirage.InitialLocation.1=500
```

- Note that the floor grid is 1 meter square.



Multiple Robots in Mirage

- Multiple robots can share the same Mirage world.
- Each must be running on a separate computer.
- You can tell Tekkotsu what machine Mirage is running on; the default is localhost.

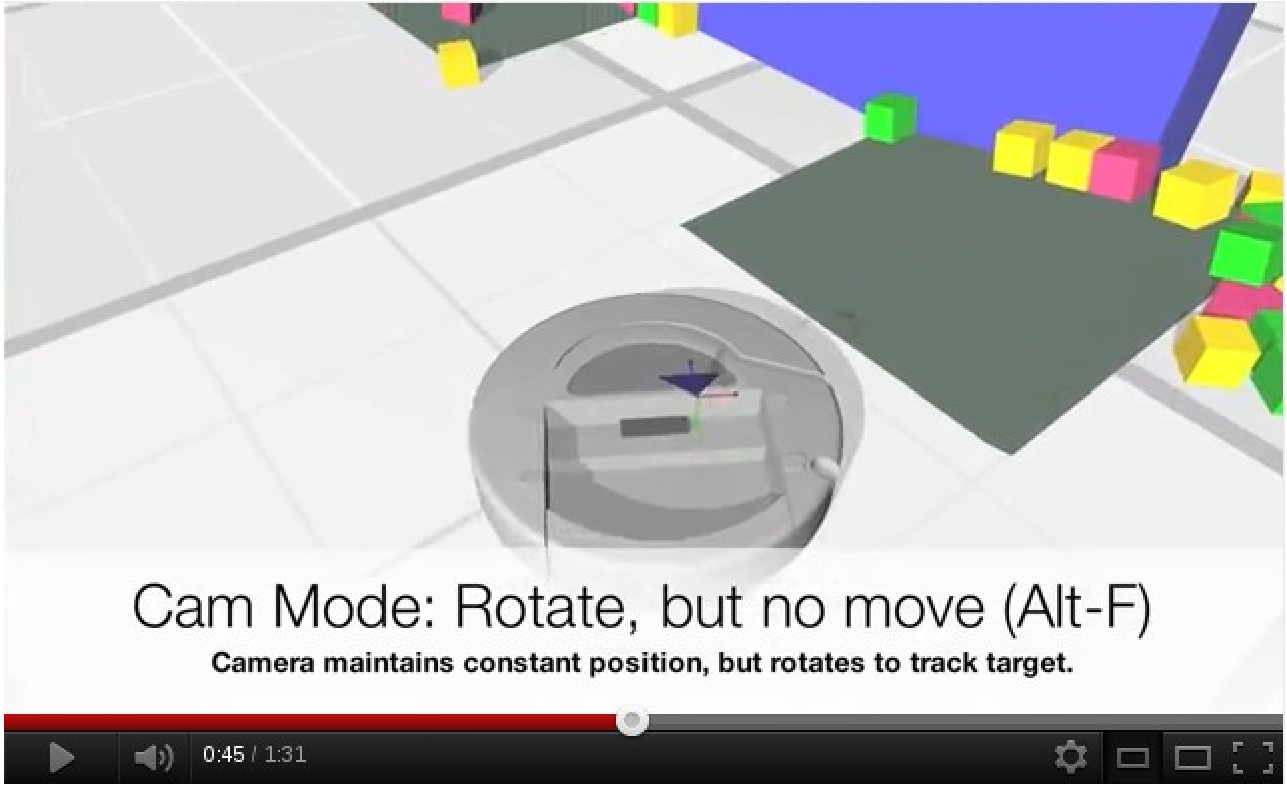
```
./tekkotsu-CREATE -c mirage.plist
     Drivers.Mirage.Host=128.2.178.7
```

Camera Tracking Modes

- By default, the camera is fixed. You move it manually.
- control-F: **mobile follow**
  - Camera follows robot but doesn't rotate.
- alt-F: **stationary follow**
  - Camera rotates to track robot, but doesn't move.
- control-alt-F: **lock to target**
  - Lock on object at center of camera and both move and rotate to track it.
- U: **un-follow** and reset view

Camera Tracking Demo Video

- Camera tracking is useful when making demo videos.



Cam Mode: Rotate, but no move (Alt-F)

Camera maintains constant position, but rotates to track target.

http://www.youtube.com/watch?v=YGwRw-Vz1zo

Mirage Tricks

- World building: edit the .ian file, run the WorldBuilder, then hit “r” in Mirage to reload the world.
- Sometimes a bug causes the camera to jump to a weird angle after a reload. Hit “u” to reset the view.
- How to make a ramp: make a cube and rotate it about the x or y axis.
- How to put a poster on a wall: make a wide but shallow cube and set the material to a texture referencing a jpg or png file.