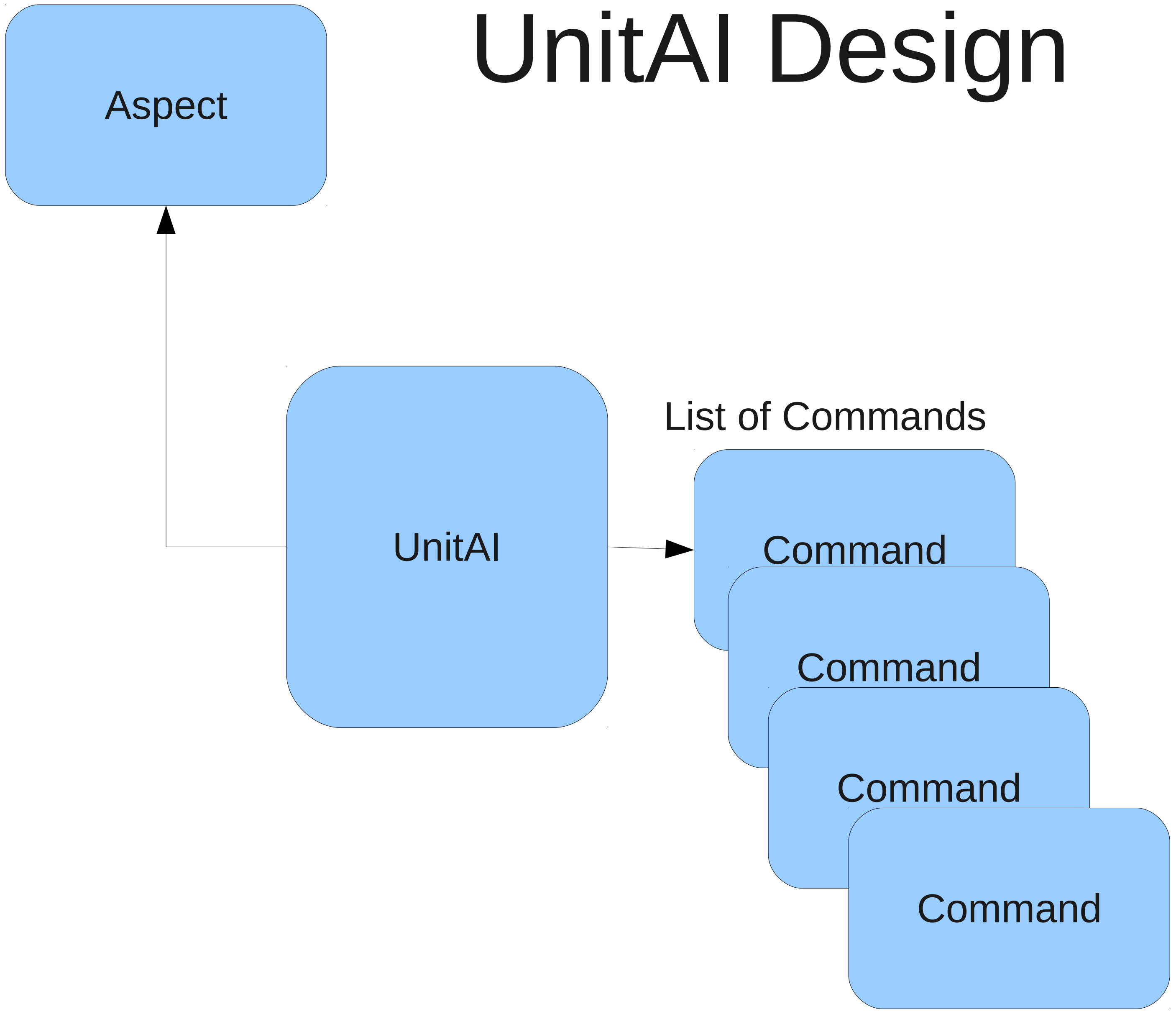
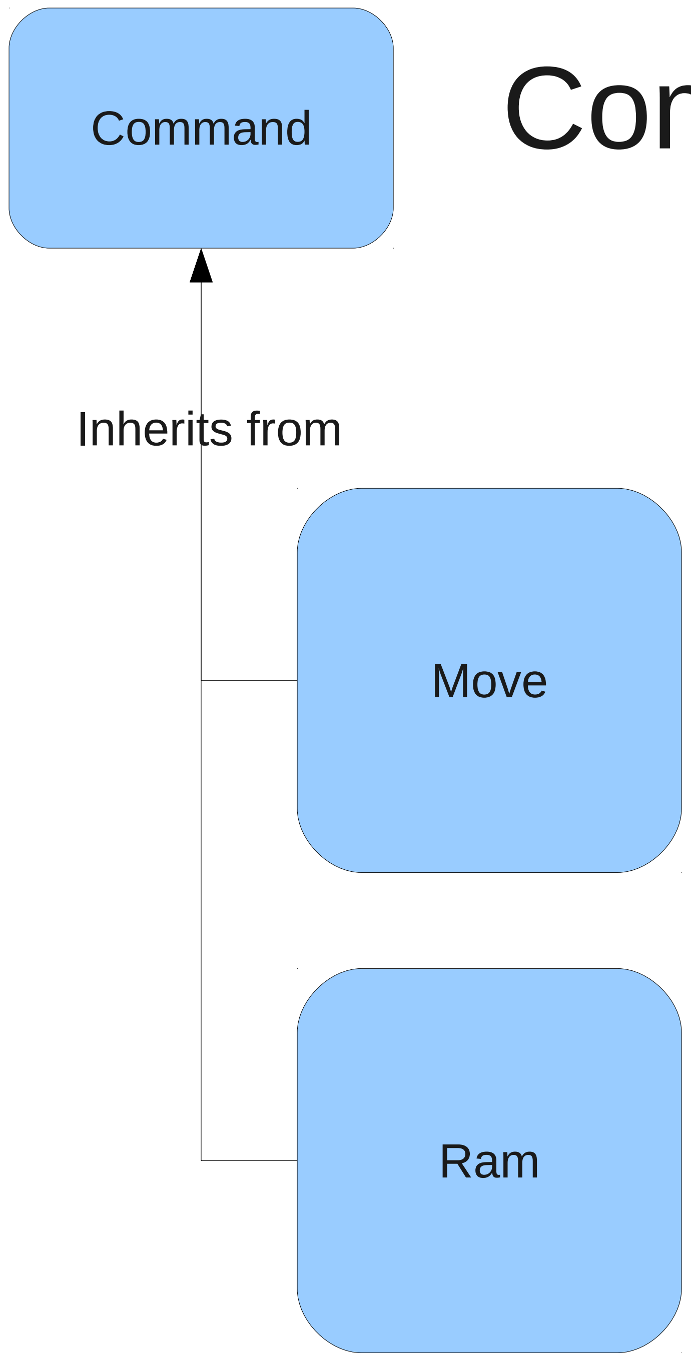
UnitAI Design

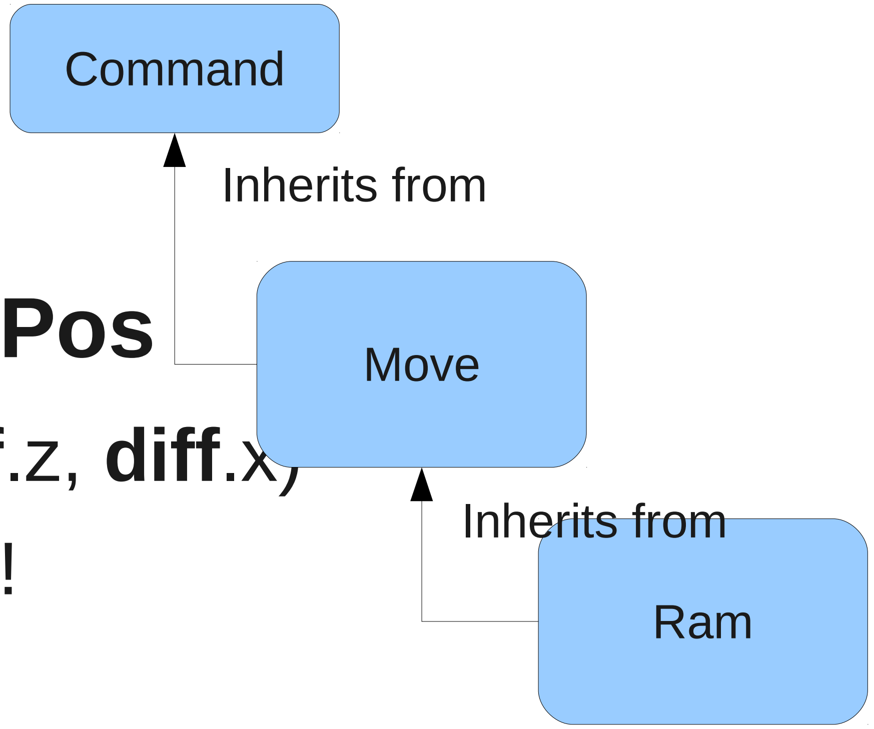


Command Design



Unit AI

- Move to a Target location
- Ram a target Entity whose location changes
- Both go towards a location – use this in inheritance
-
- **diff** = **targetLocation** - **myPos**
  - desiredHeading = atan2(**diff**.z, **diff**.x)
  - desiredSpeed   = maximum!



Unit AI 2

- Predict moving target location
- Where will the target be?
- Look at relative speed = |**targetVel** – **myVel**|
- Look at distance = |**diff**| from last slide
- Time 't' to travel distance = distance/speed
- Predicted target location = **targetVel** * t

Mouse Selection

```
#self.ms is mouse state (mouse.getMouseState())

self.ms.width = self.engine.gfxSystem.viewport.actualWidth
self.ms.height = self.engine.gfxSystem.viewport.actualHeight
self.mousePos = (self.ms.X.abs/float(self.ms.width), self.ms.Y.abs/float(self.ms.height))
mouseRay = self.engine.cameraSystem.camera.getCameraToViewportRay(*self.mousePos)
result  =  mouseRay.intersects(self.groundPlane)

if result.first:
    pos =  mouseRay.getPoint(result.second)
    self.mousePosWorld = pos
```

Pos   is the location in the x-z plane corresponding to where you clicked your mouse. The entity nearest this position that is within a threshold distance squared Is the selected entity. Indicated by an ogre AABB

Group Selection

- Intermediate tutorial
- All selected entities should be indicated by an ogre AABB
- Commands applied to ALL selected entities

OIS and Mouse Cursors

// insert right before calling mInputSystem = OIS::InputManager::createInputSystem(paramList);

- #if defined OIS_WIN32_PLATFORM
- paramList.insert(std::make_pair(std::string("w32_mouse"), std::string("DISCL_FOREGROUND")));
- paramList.insert(std::make_pair(std::string("w32_mouse"), std::string("DISCL_NONEXCLUSIVE")));
- paramList.insert(std::make_pair(std::string("w32_keyboard"), std::string("DISCL_FOREGROUND")));
- paramList.insert(std::make_pair(std::string("w32_keyboard"), std::string("DISCL_NONEXCLUSIVE")));
- #elif defined OIS_LINUX_PLATFORM
- paramList.insert(std::make_pair(std::string("x11_mouse_grab"), std::string("false")));
- paramList.insert(std::make_pair(std::string("x11_mouse_hide"), std::string("false")));
- paramList.insert(std::make_pair(std::string("x11_keyboard_grab"), std::string("false")));
- paramList.insert(std::make_pair(std::string("XAutoRepeatOn"), std::string("true")));
- #endif