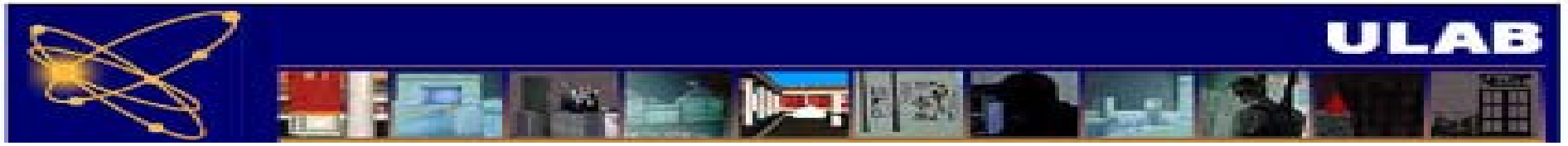
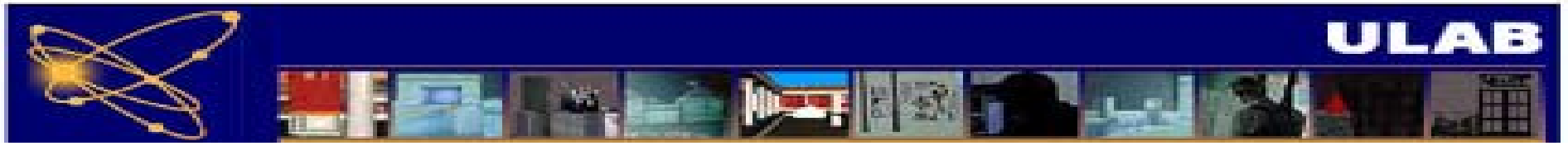


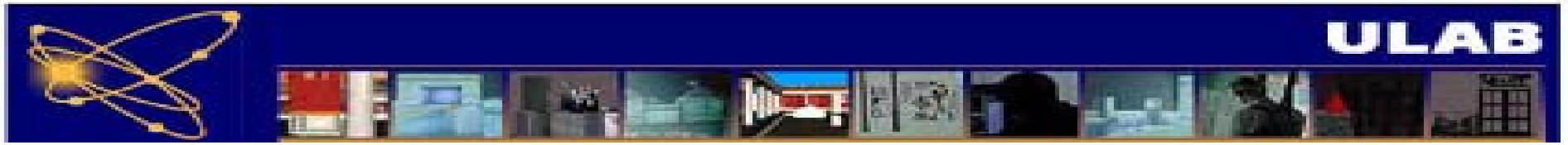
USARSim Tutorial

Basic Session



Outline

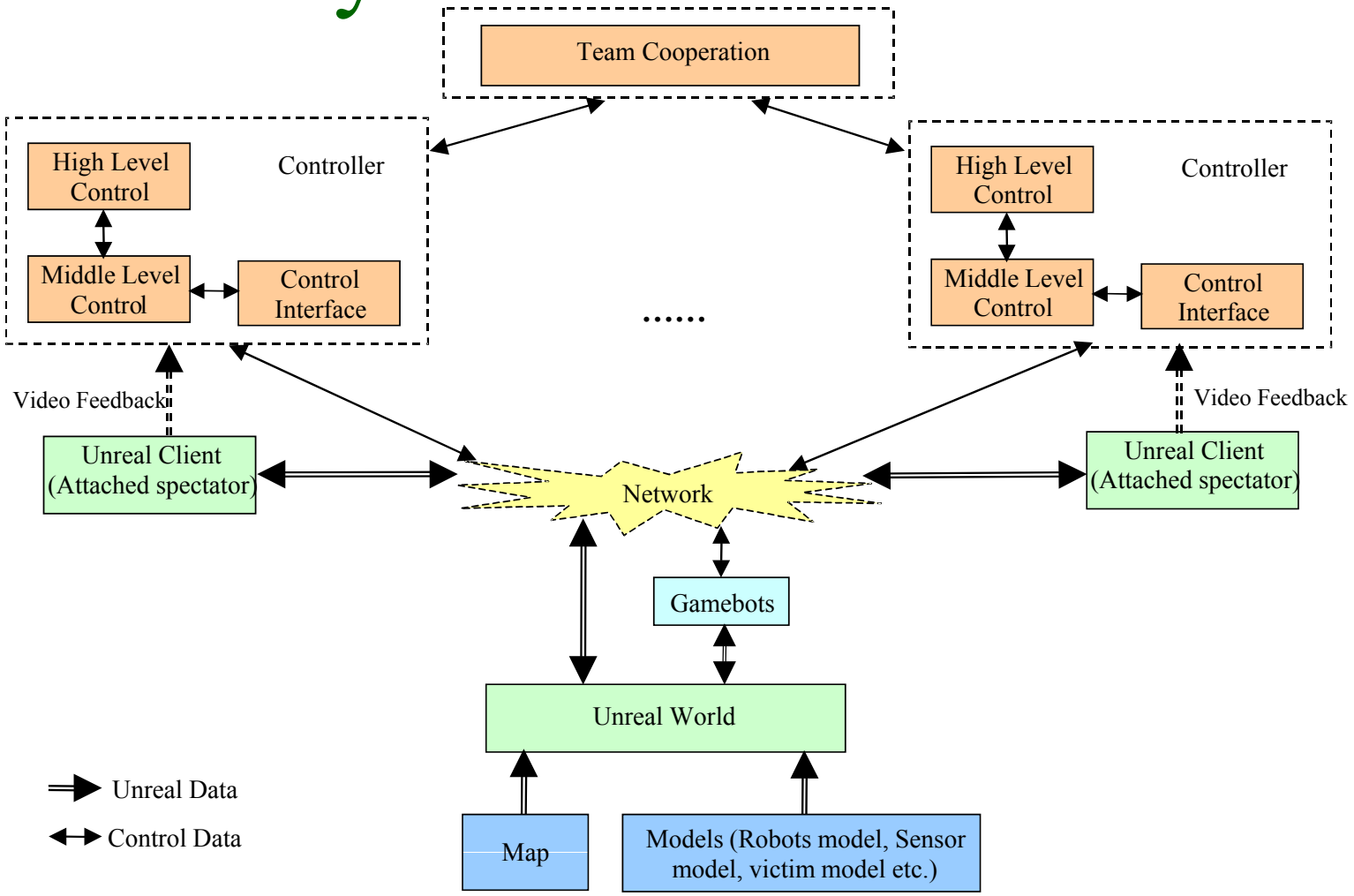
- Introduction
- System Architecture
- Simulator Components
  - Environment simulation
  - Sensor simulation
  - Robot simulation
  - Control simulation
- Using USARSim



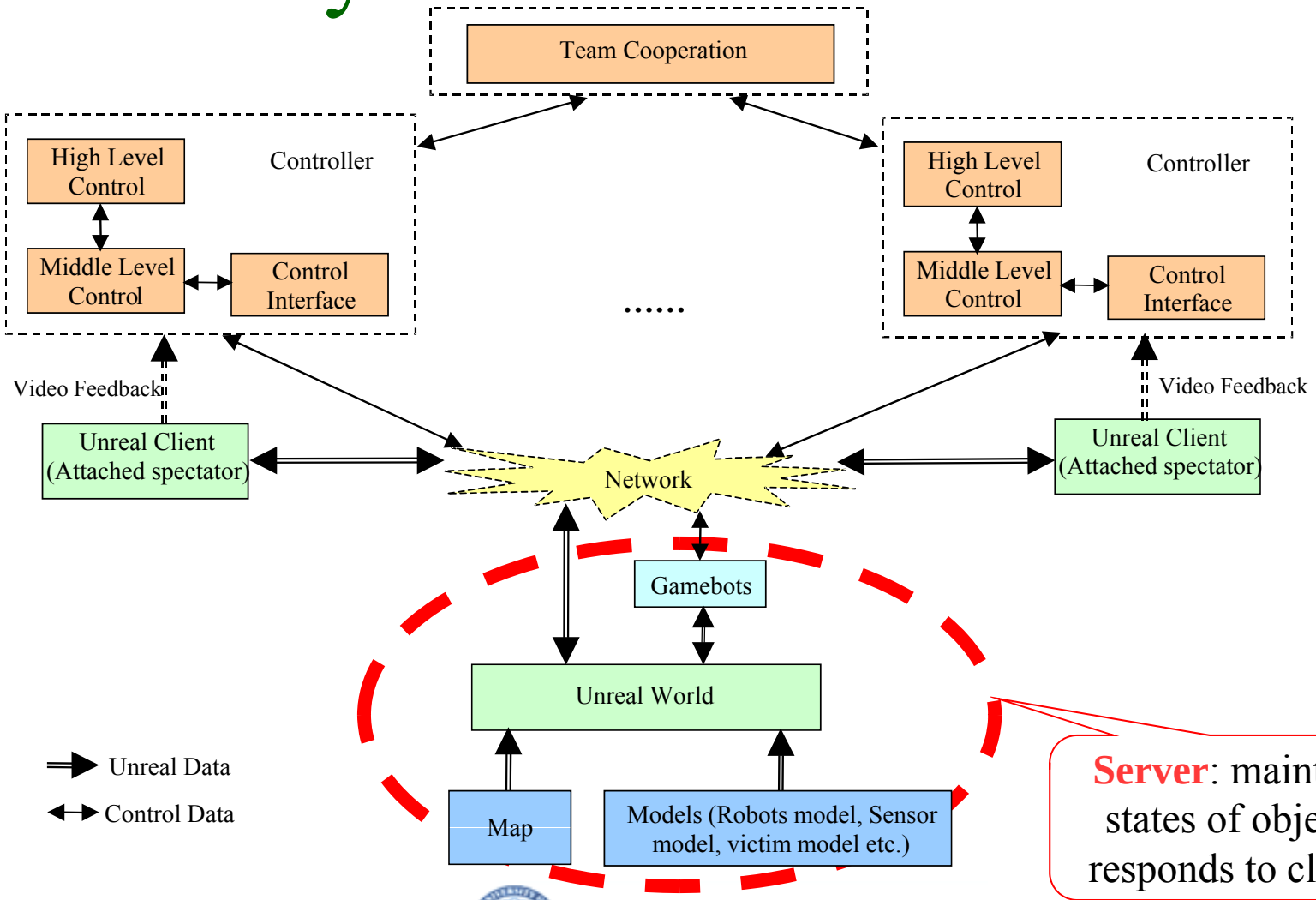
Introduction

- What is USARSim
  - Game based high fidelity interactive simulation of urban search and rescue (USAR) robots and environments
  - Provides:
    - environmental models (levels) of the NIST Yellow, Orange, and Red Arenas
    - robot models of commercial and experimental robots
    - sensor models
    - auxiliary tools for robot control
  - Research tool for the study of human-robot interaction (HRI) and multi-robot coordination.

System Architecture

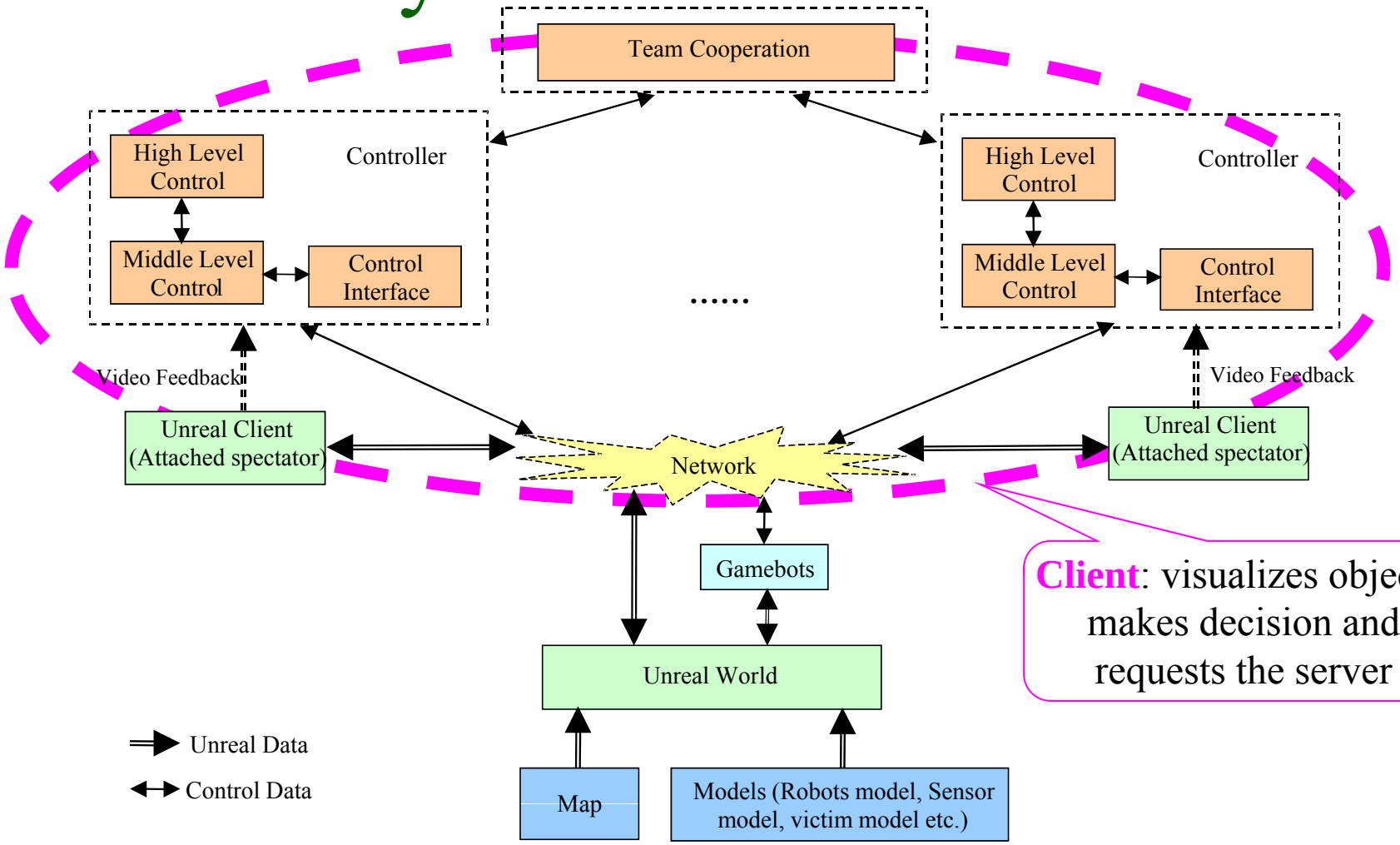


System Architecture



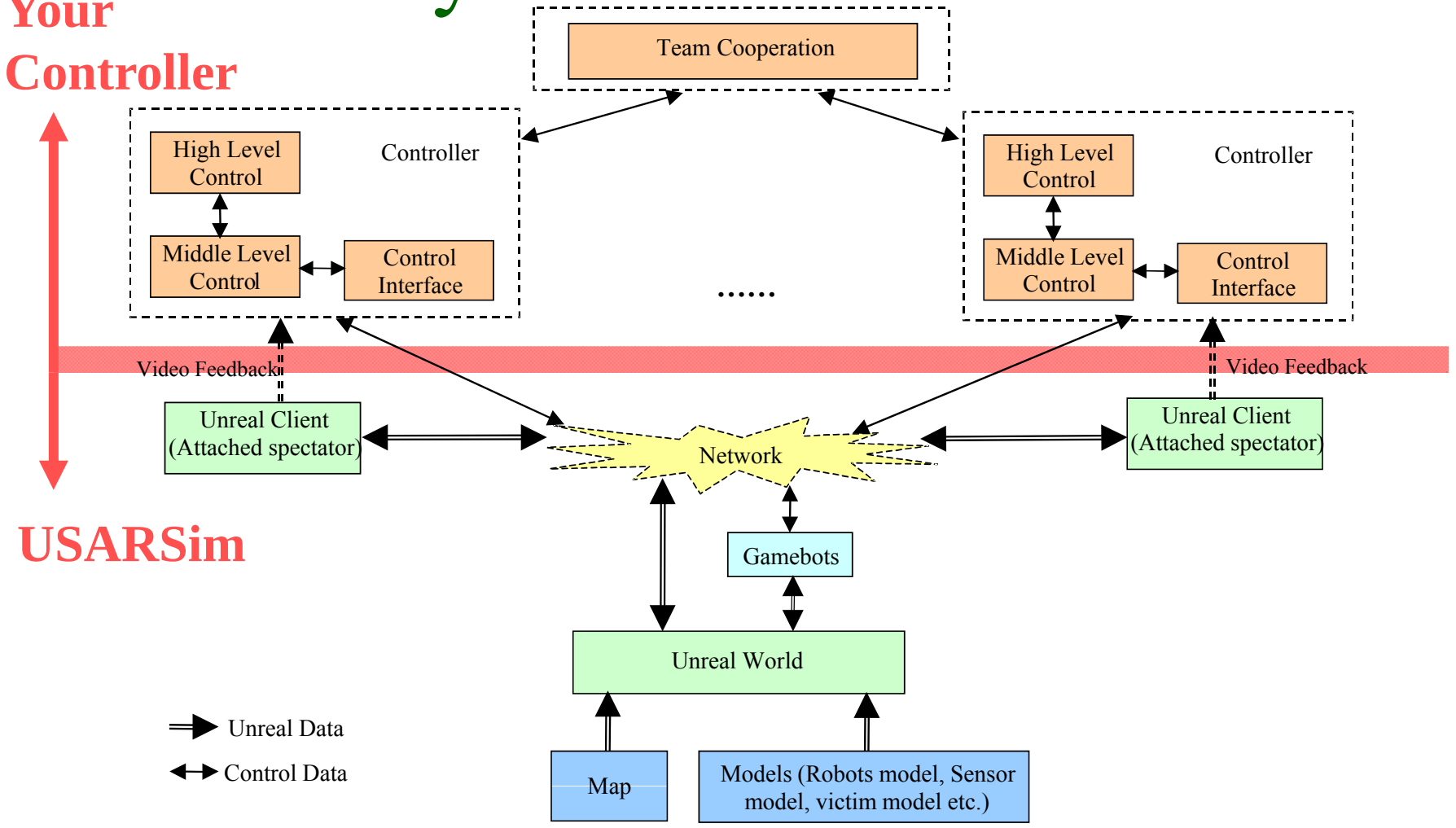
Server: maintains states of objects, responds to clients

System Architecture

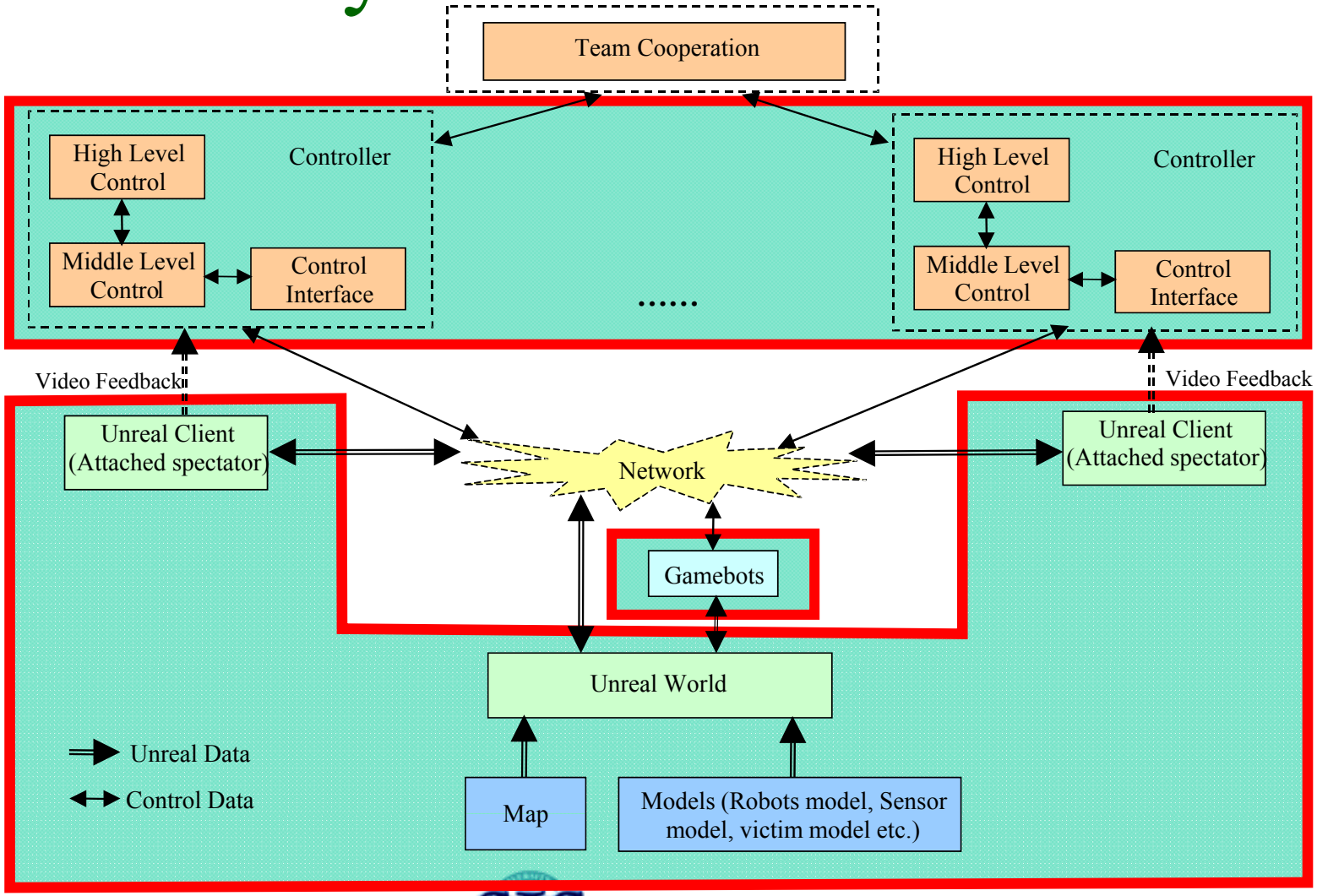


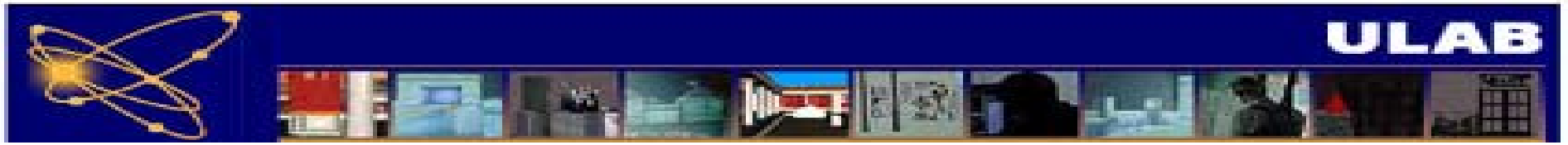
Client: visualizes objects, makes decision and requests the server

System Architecture



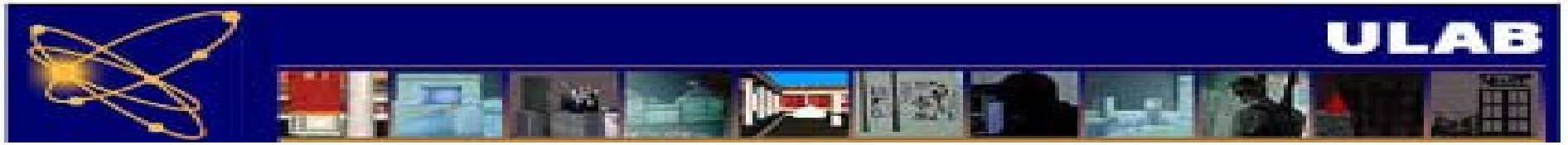
System Architecture





System Architecture (cont.)

- Unreal Engine
  - A multiplayer combat-oriented first-person game engine released by Epic Games
    - *3D scene render*
    - *Physic engine (Karma engine)*
    - *Script language*
    - *3D authoring tool*
- Gamebots
  - A modification to Unreal Tournament to bridge Unreal engine with the outside applications
    - *TCP socket connection*
    - *Messages exchange*
- Controller
  - The application designed by users for research purposes
    - *Robot control*
    - *Data exchange*

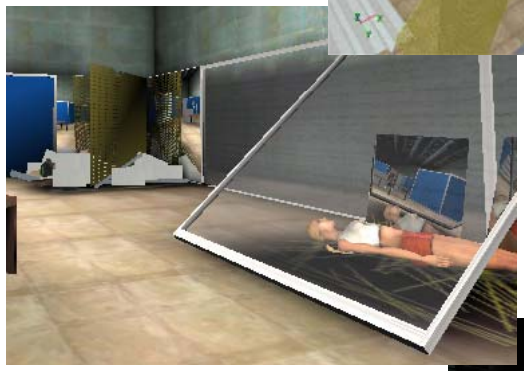


Simulator Components

- Components
  - Environment simulation
  - Sensor simulation
  - Robot simulation
  - Control simulation

Environment Simulation

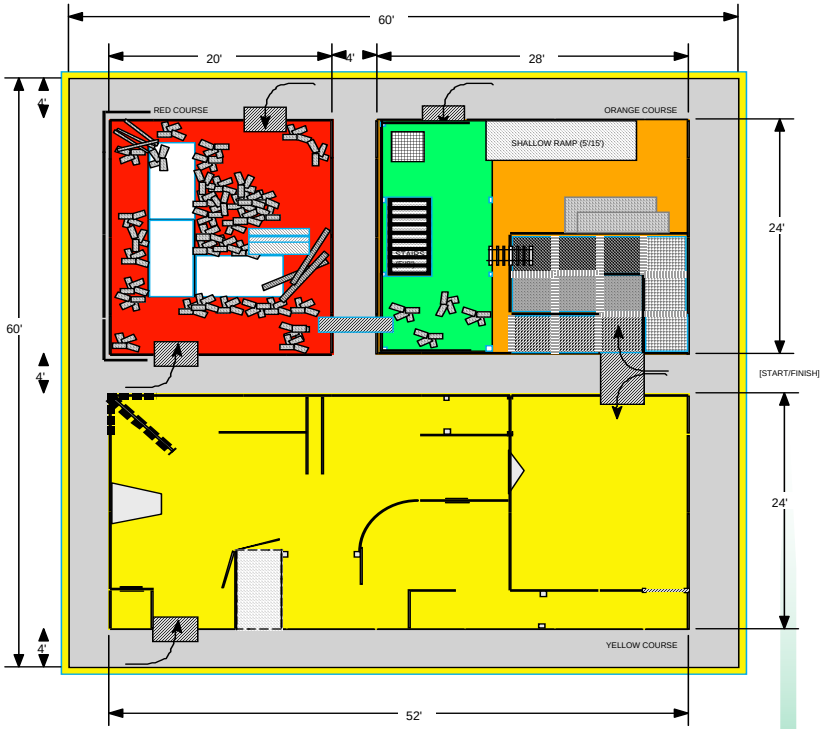
- Components
  - Geometric models
  - Obstacles
  - Light
  - Special effects
  - Victim
- NIST Arenas
  - The real arenas



Environment Simulation (cont.)

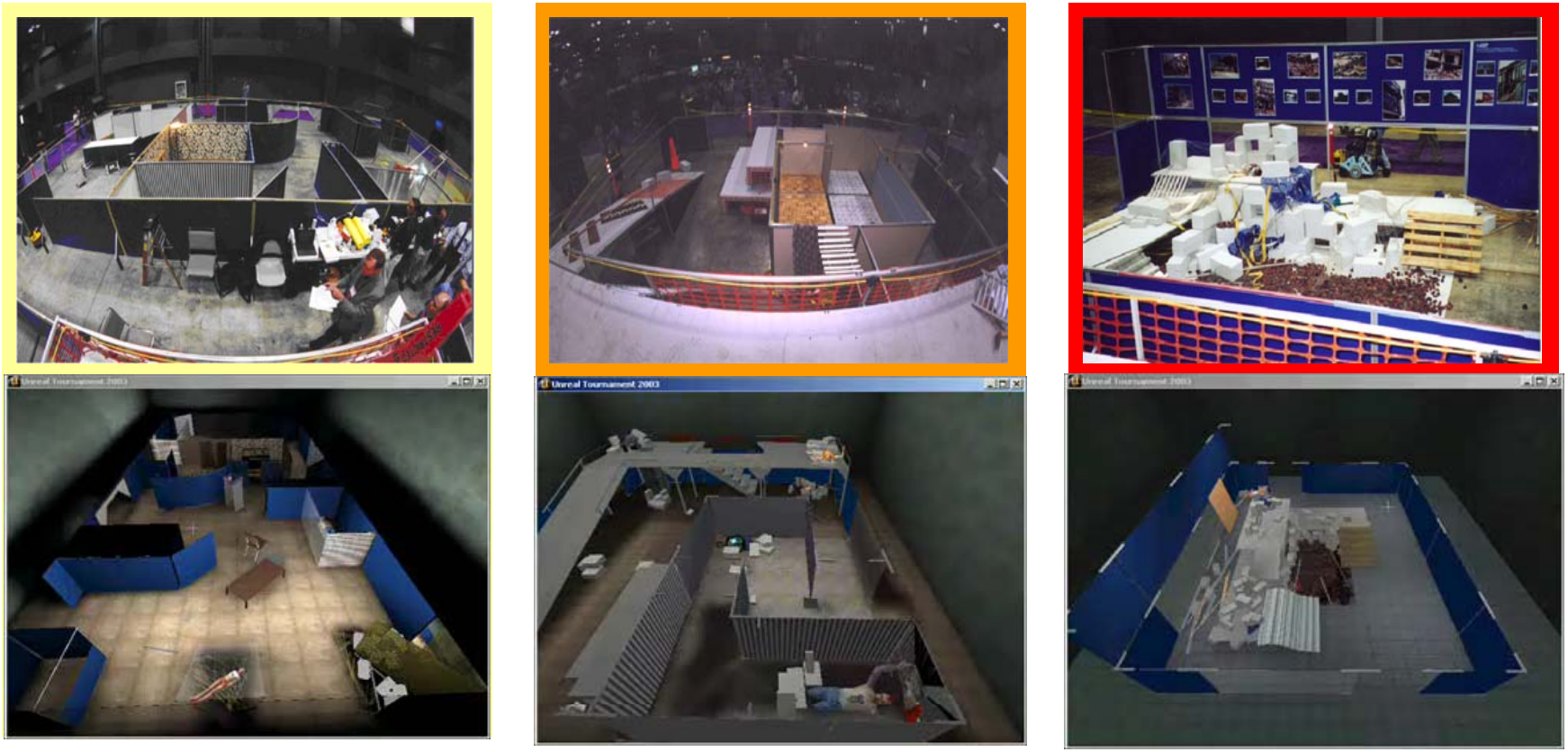
(from Jacoff et al. 2003)

- **Yellow Arena**
  - Simple to traverse, no agility requirements
  - Planar (2-D) maze
  - Isolates sensors with obstacles/targets
  - Reconfigurable in real time to test mapping
- **Orange Arena**
  - More difficult to traverse, variable floorings
  - Spatial (3-D) maze, stairs, ramp, holes
  - Physical obstacles include rubble, paper, pipes
  - Similarly reconfigurable
- **Red Arena**
  - Difficult to traverse, unstructured environment
  - Simulated rubble piles, shifting floors
  - Problematic junk (rebar, plastic bags, pipes…)



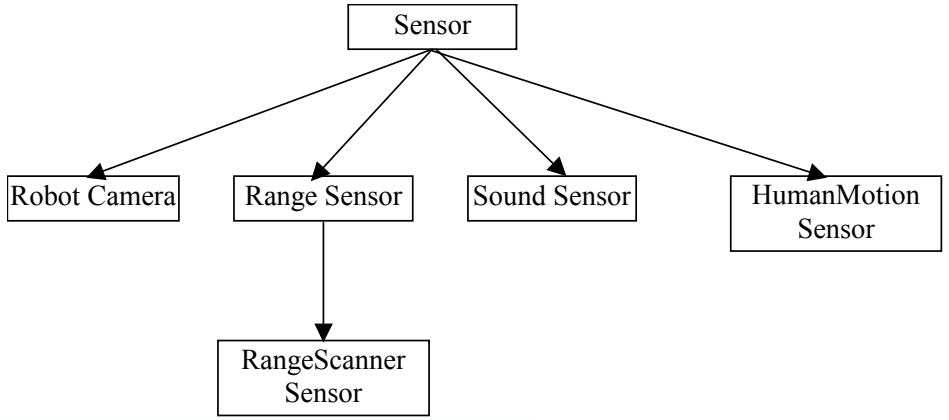
Environment Simulation (cont.)

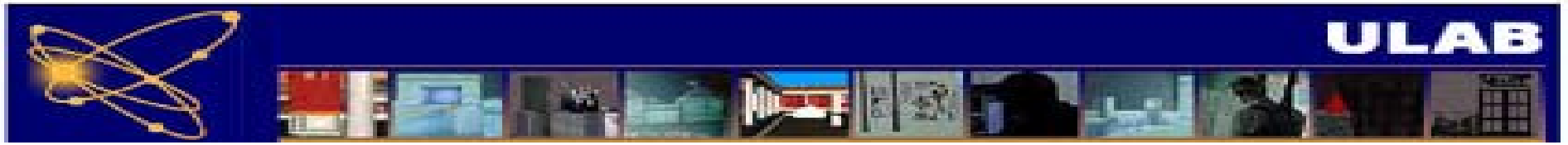
- The simulated arenas



Sensor Simulation

- Method
  - Calculate data from the ground truth database
  - Add data noise and distortion
- Features
  - Hierarchical structure
  - Configurable





Sensor Simulation (cont.)

- Sensors
  - State sensors
    - Battery state, headlight state, location, rotation, velocity sensors etc.
  - Perception sensors
    - sonar, laser and pan-tilt-zoom (ptz) camera
- Video feedback

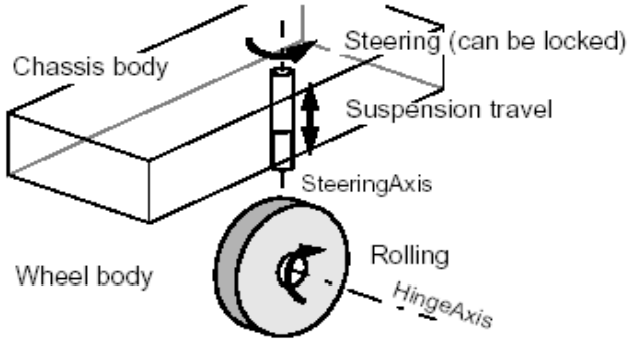
  Attach viewpoint to the camera in Unreal Client
  - Web Camera
    - Capturing scenes in Unreal Client
    - Send out pictures through network
      - Raw image, jpeg image

Robot Simulation

- The Robot model

  Karma engine based configurable model

  - Features
    - Encapsulate the programming details
    - Building robot by assembling
  - Components
    - Chassis
    - Joints
    - Parts
    - Attached auxiliary items
  - Method
    - Connect chassis and joints through joints
    - Attach auxiliary items to chassis or parts



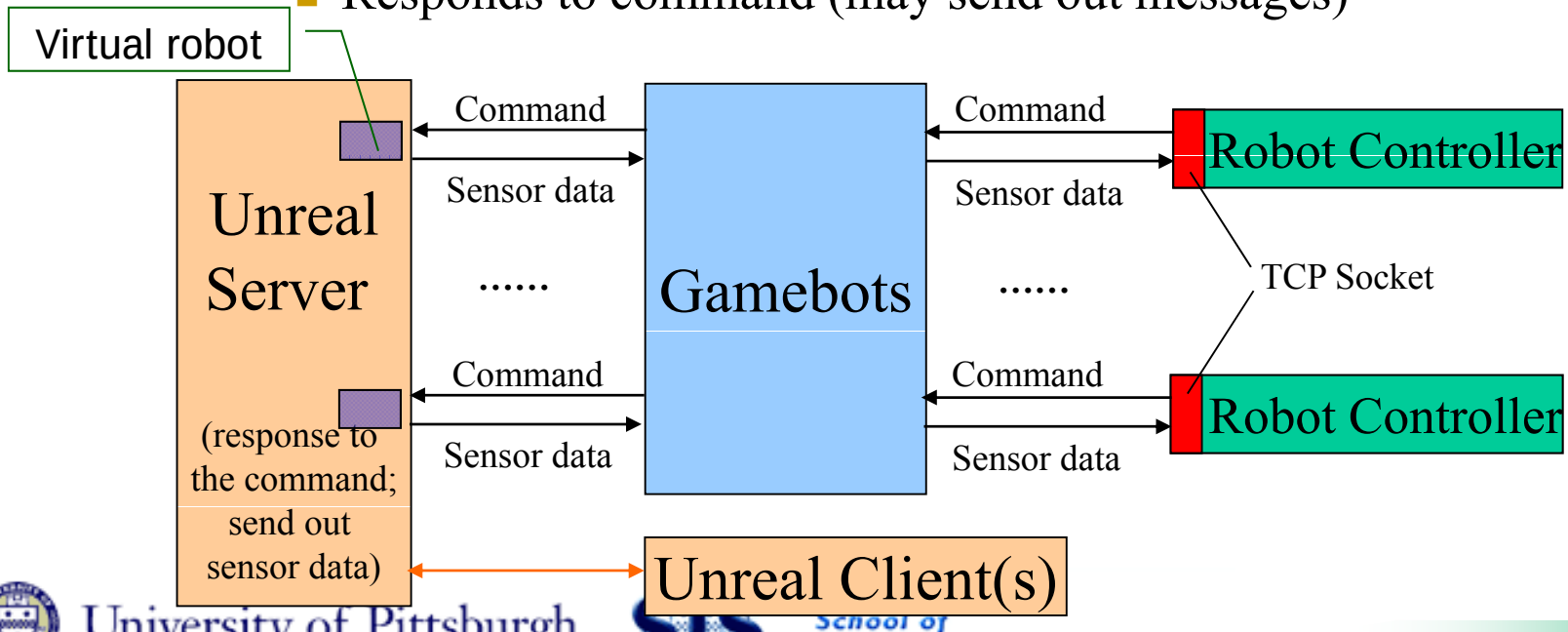
Robot Simulation (cont.)

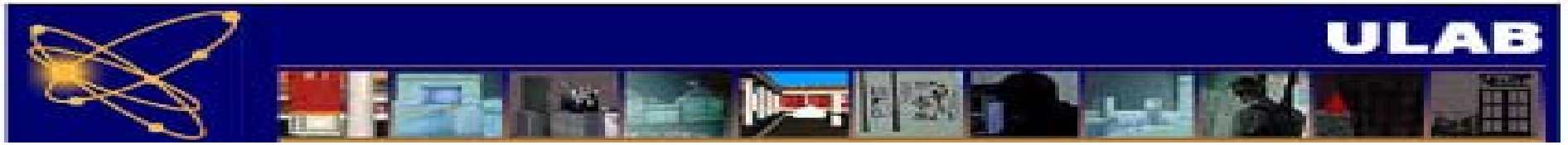
- The robots

Real robots vs. simulated robots

Control Simulation

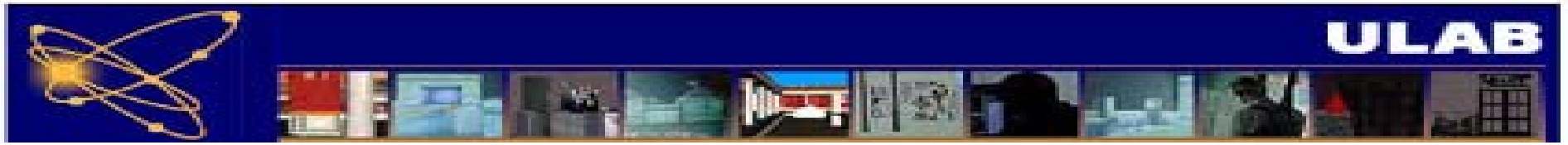
- Method
  - Gamebots
    - communicates between controller and robot
  - Server (the virtual robot)
    - Consistently sends out messages
    - Responds to command (may send out messages)





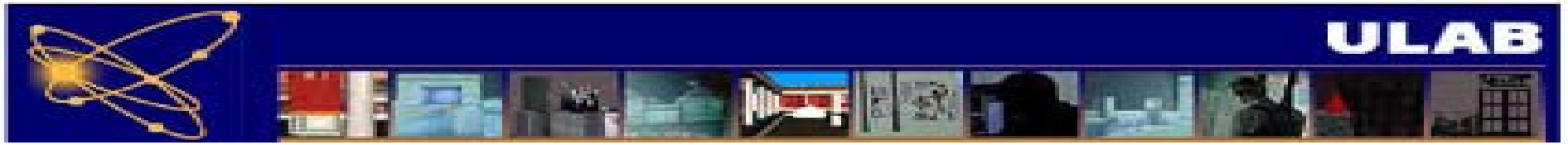
Control Simulation (cont.)

- Communication data
  - Messages
    - State message
    - Sensor messages
    - Geometry and configuration messages
  - Commands
    - Robot spawning command
    - Wheel/joint control commands
    - Camera control command
    - Query commands
- Auxiliary tools
  - Pyro with USARSim plug-in



Control Simulation (cont.)

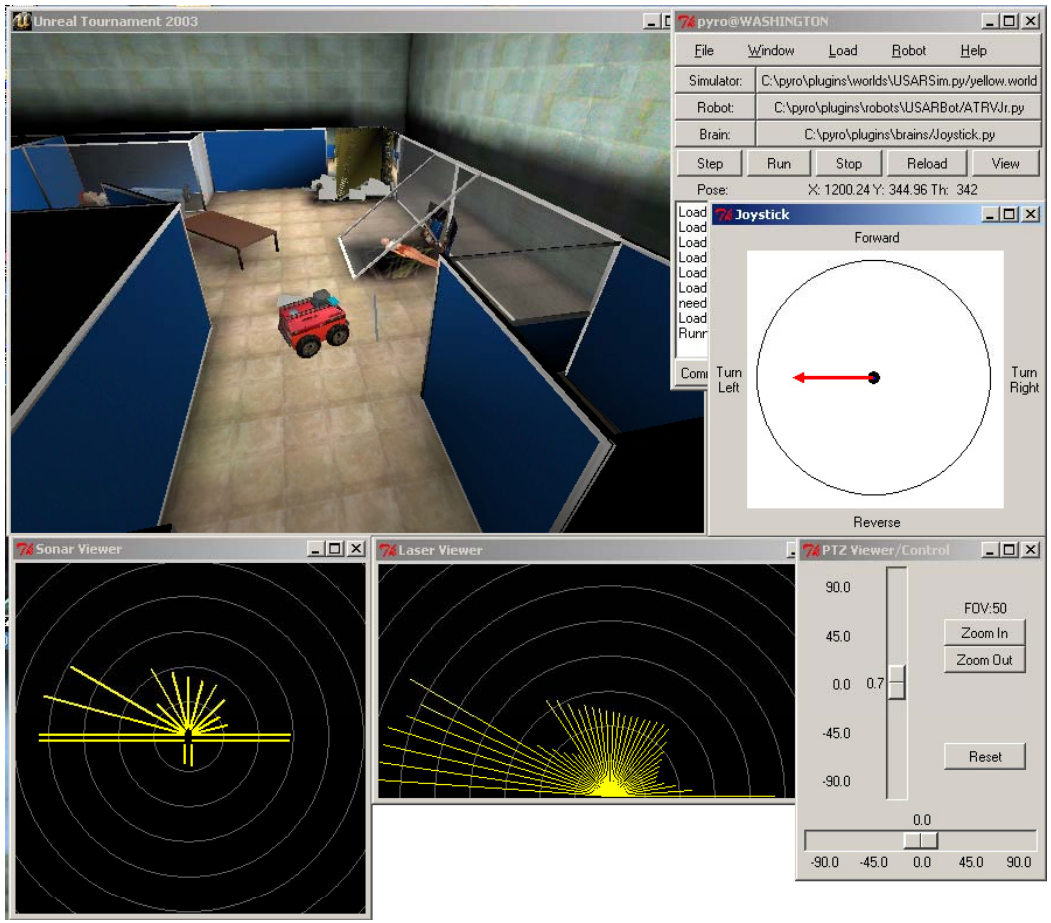
- Pyro
  - *"a Python library, environment, GUI, and low-level drivers used for explore AI and robotics"*
- USARSim plug-in
  - Abstract USARSim robot to pyro.robot
  - Share the same controller and GUI
- Player with USARSim drivers
  - Player
    - *"a robot device server that gives users simple and complete control over the sensors and actuators on the robot"*
  - USARSim drivers
    - encapsulates lower-level details
    - Use USARSim devices as physical devices



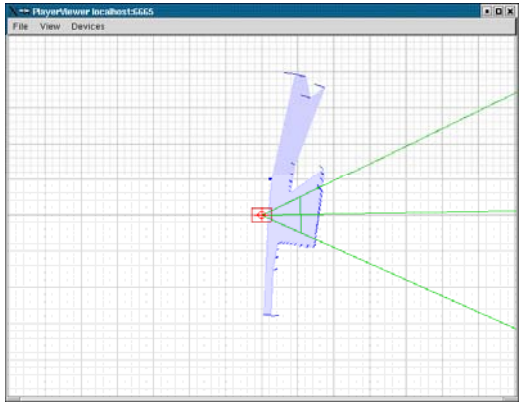
Using USARSim

- Installation
  - Unreal Tournament 2003 (UT2003) & 2225 patch
    - Windows
    - Linux (*when it asks for the disk 1, try disk 2.*)
  - Unreal Engine 2 Runtime (UE2RT)
    - Windows (http://udn.epicgames.com/Two/UnrealEngine2Runtime22262001)
- Viewpoint control
  - *Left* mouse button – switch viewpoint
  - *Right* mouse button – release attached viewpoint
  - '*C*' – overlooking effect
- Using off-the-shelf tools
  - Pyro with USARSim plug-in
  - Player with USARSim drivers

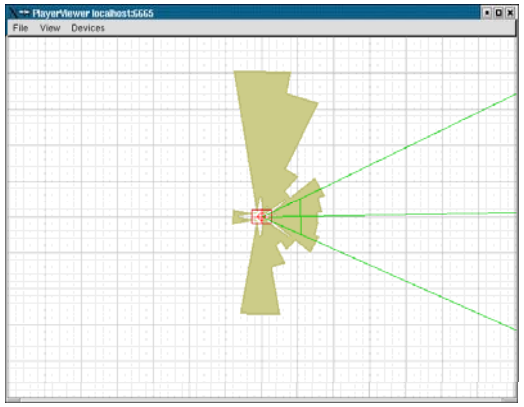
Using USARSim (cont.)



Pyro with services and brain



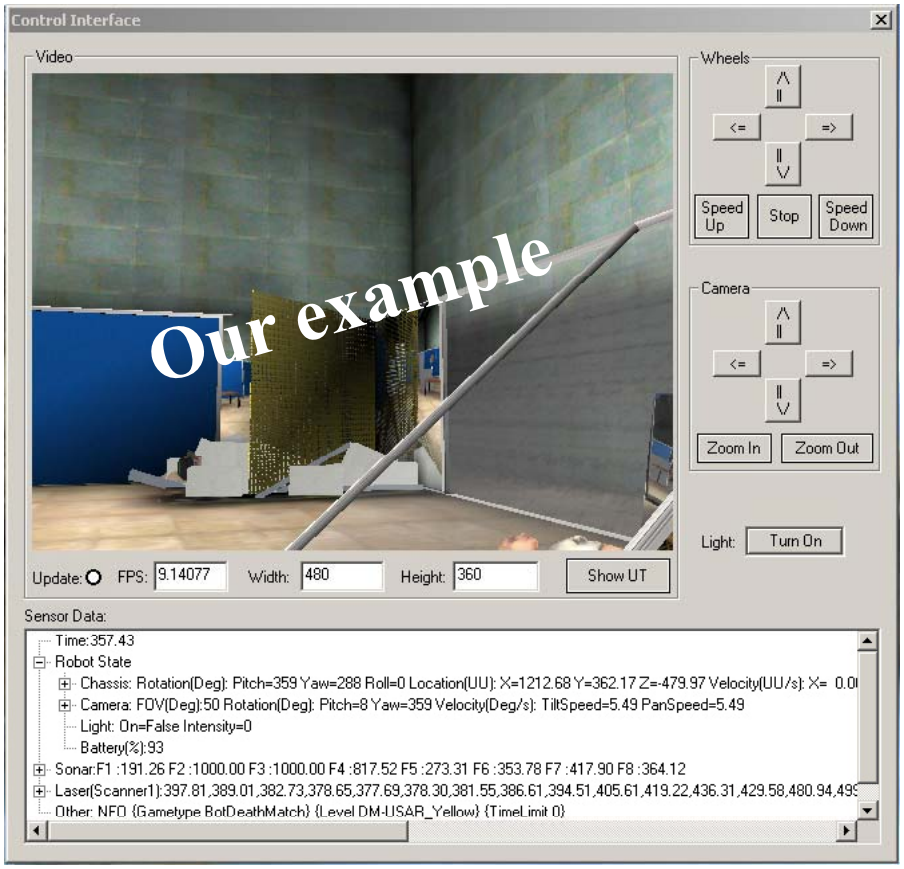
Laser sensor on Playerv



Sonar sensor on Playerv

Using USARSim (cont.)

- Build your own controller



Your Controller !