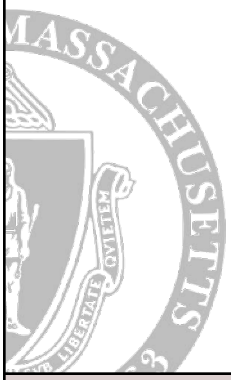
UMassAmherst

Roger-the-Crab



Rod Grupen
Department of Computer Science
University of Massachusetts Amherst

UMassAmherst

Un-Crating your Roger - C and X windows

Platforms:

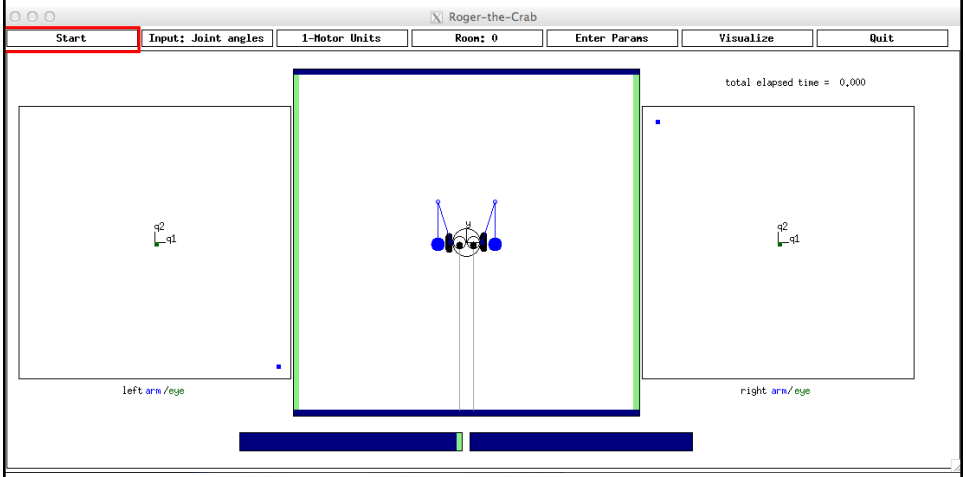
- Linux (VM, dual booting works), OS X (+Xquartz)

Getting Started:

- download your copy of the Roger simulator from
  www-robotics.cs.umass.edu/~grupen/403/code
- unpack the compressed tar/zip file:
  tar -xvf roger-2020.tar
- follow the directions in the README

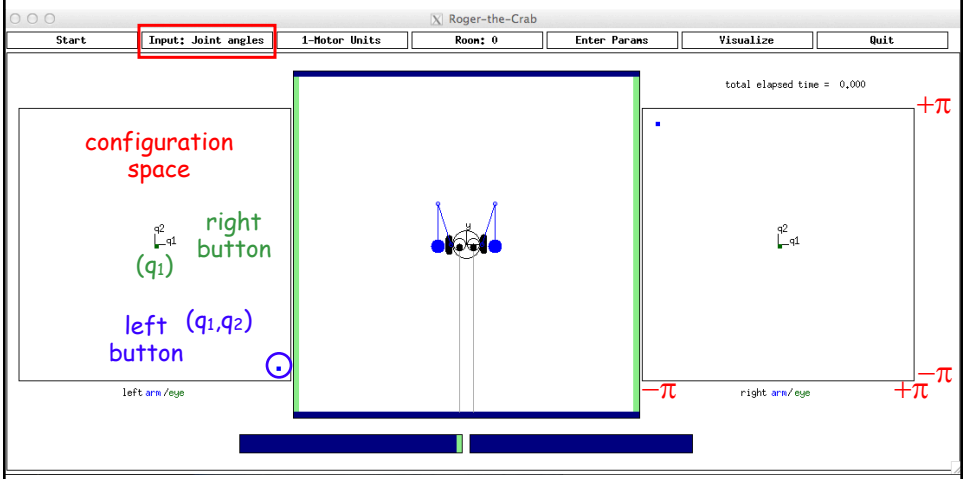
UMassAmherst

The Simulator (Development) Environment



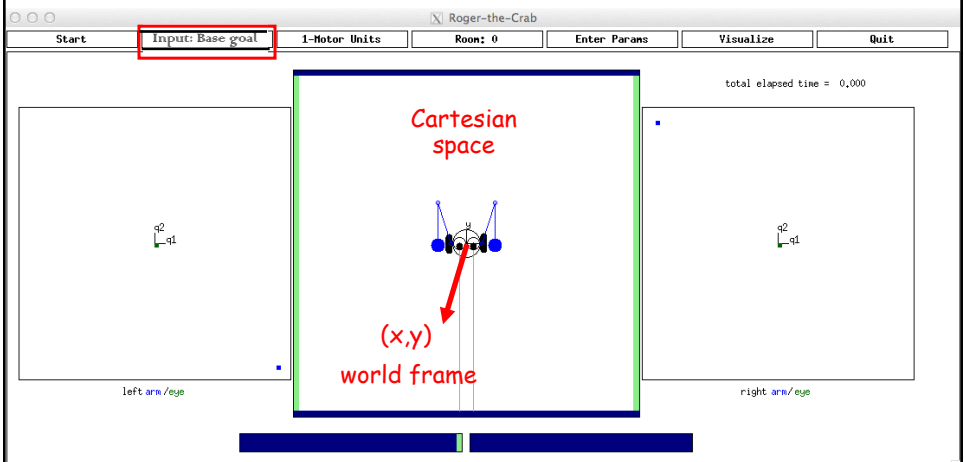
UMassAmherst

Input Modes: Joint Angle inputs



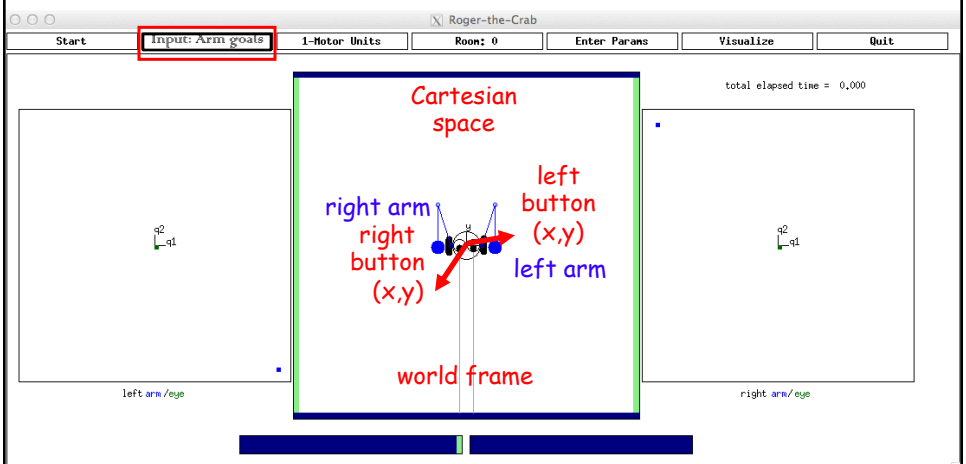
UMassAmherst

Input Modes: Base goals



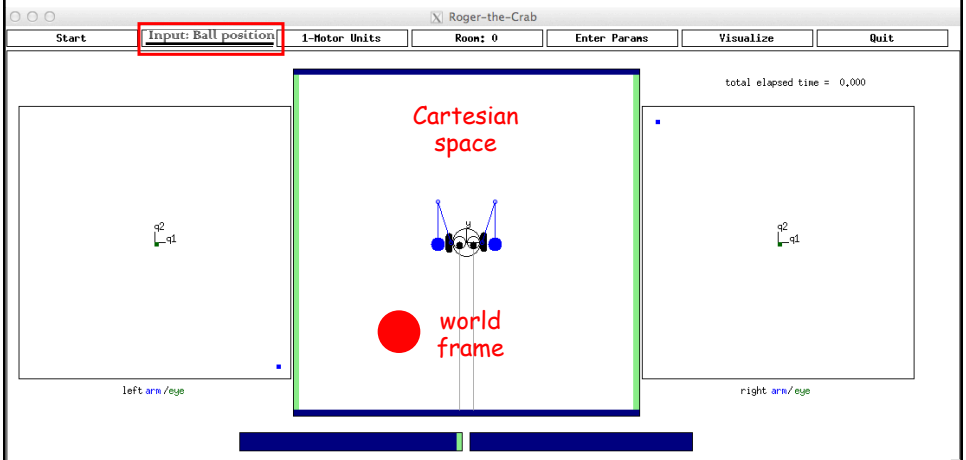
UMassAmherst

Input Modes: Arm goals



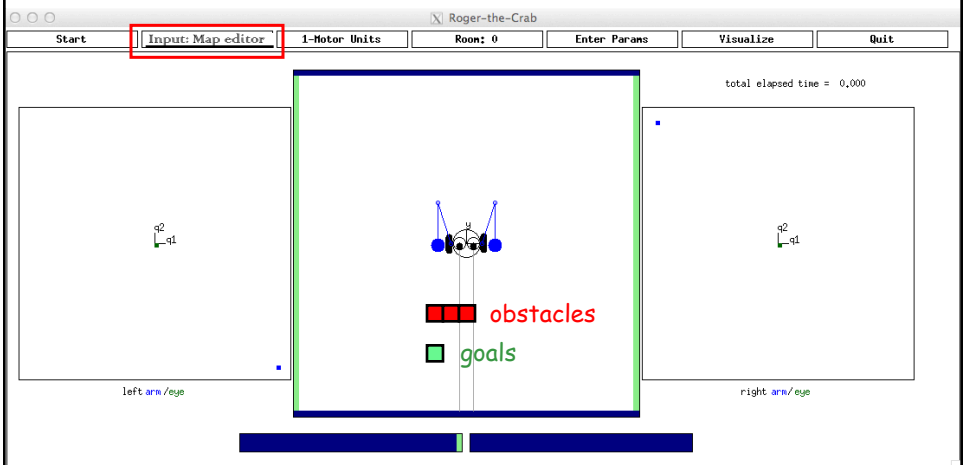
UMassAmherst

Input Modes: Introducing an Object (Ball)



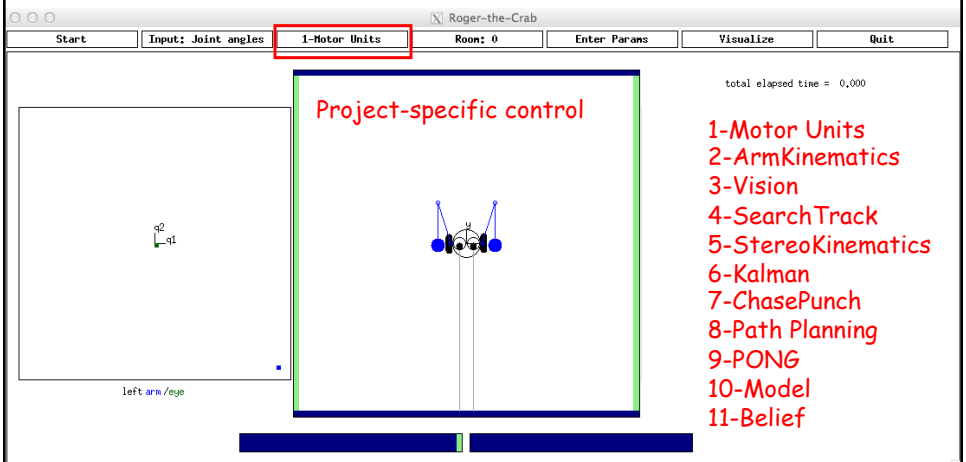
UMassAmherst

Input Modes: Map Editor



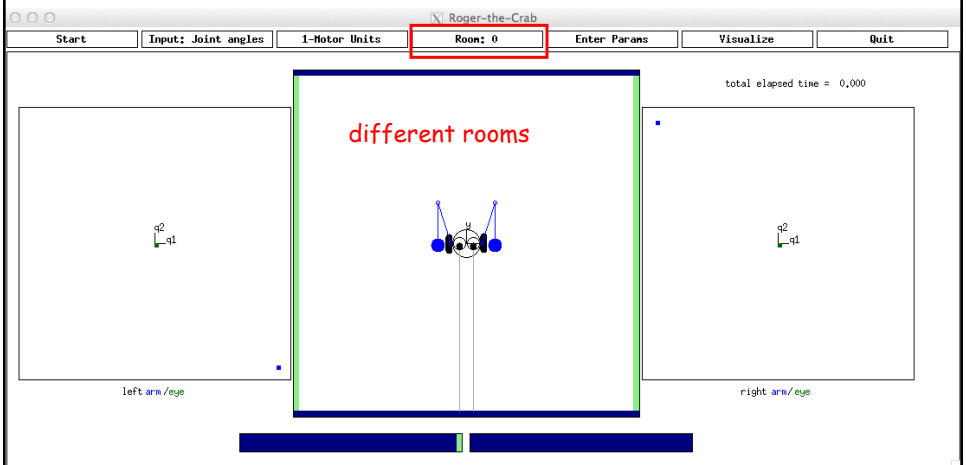
UMassAmherst

Control Modes



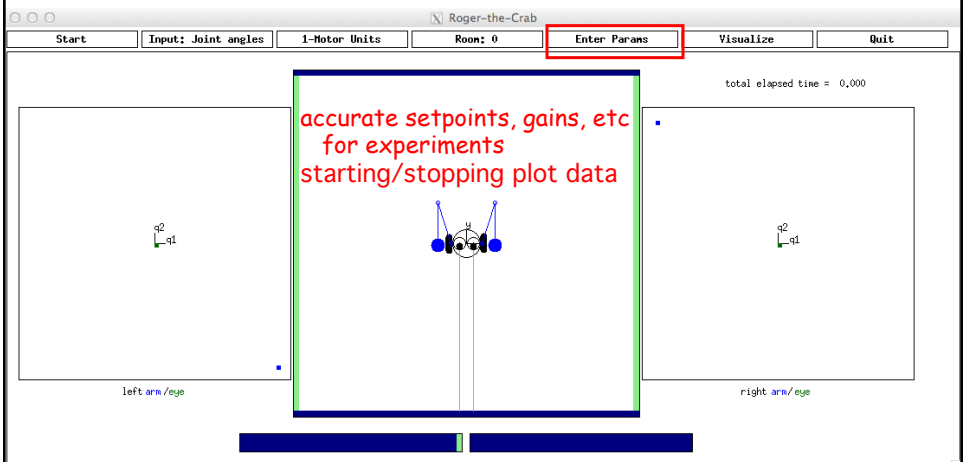
UMassAmherst

Environmental Maps



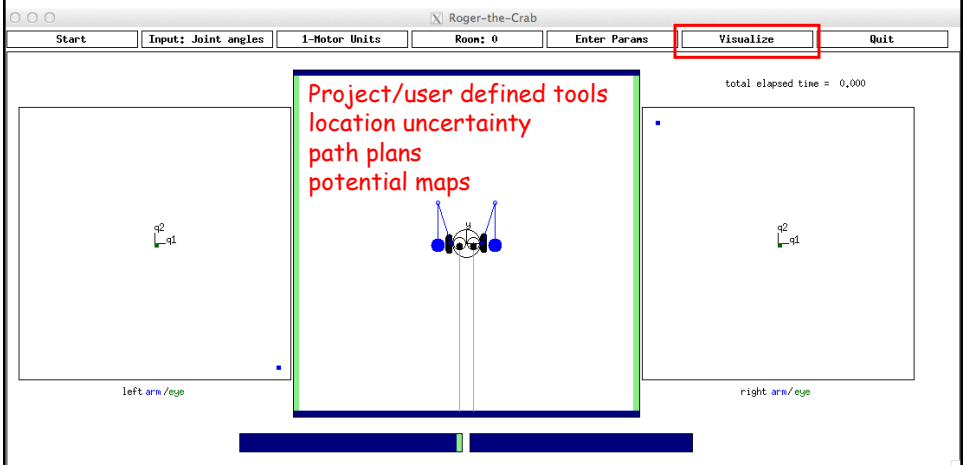
UMassAmherst

Command line I/O

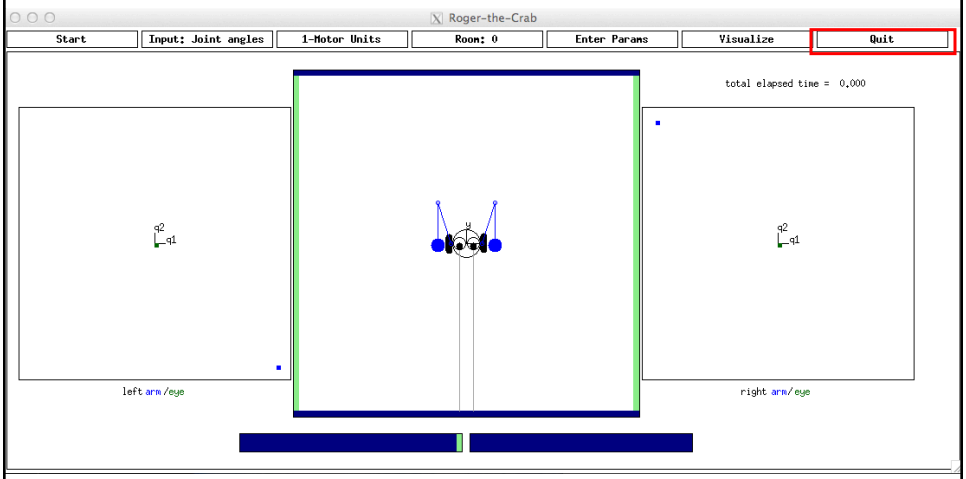


UMassAmherst

Project Specific Visualization

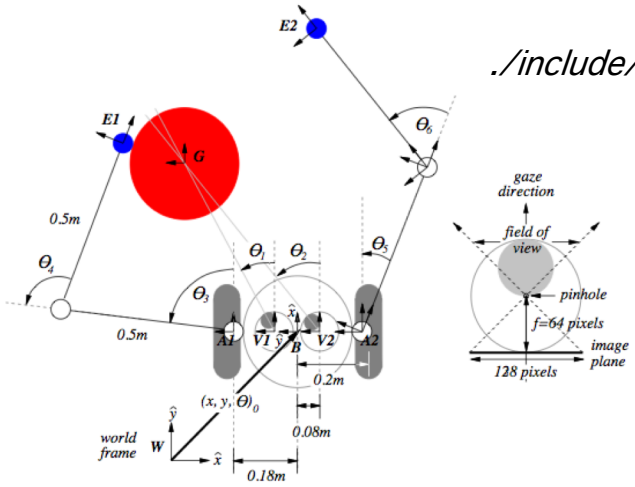


Quiting the Simulator



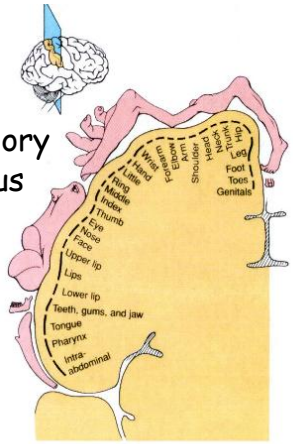
Roger-the-Crab - Kinematic Definition

./include/roger.h

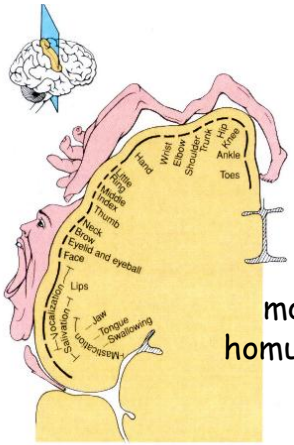


Primary Cortical Homunculi

somatosensory homunculus



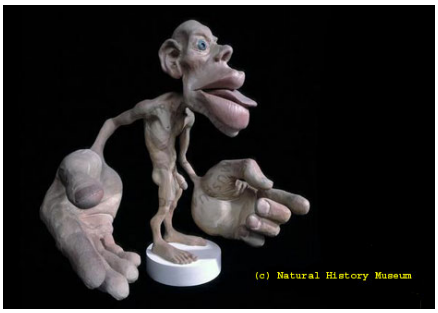
(a) Somatosensory cortex in right cerebral hemisphere



(b) Motor cortex in right cerebral hemisphere

motor homunculus

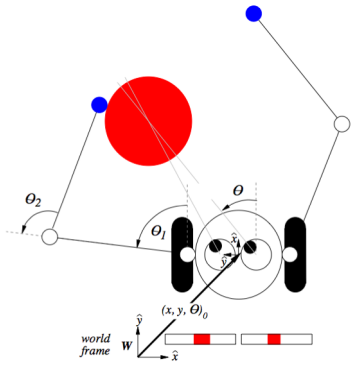
Homunculus Man



somatosensory homunculus

motor homunculus

Afferents



eyes:

- $\theta[2]$, $\dot{\theta}[2]$, images[2][128][3]

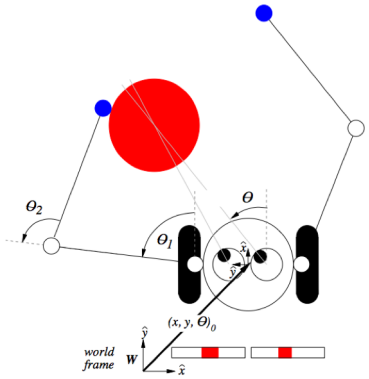
arms:

- $\theta_1[2]$, $\dot{\theta}_1[2]$, $\theta_2[2]$, $\dot{\theta}_2[2]$
- tactile (force) sensors: $f[2] \in R^2$

mobile base:

- position (x, $\dot{x}$, y, $\dot{y}$), orientation (θ, $\dot{\theta}$)
- bump (force) sensor: $\boldsymbol{f} \in R^2$

Efferents



eye torques:

- $\tau[2]$

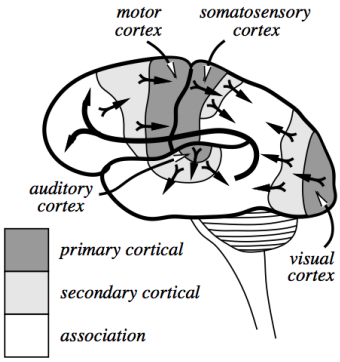
arm torques:

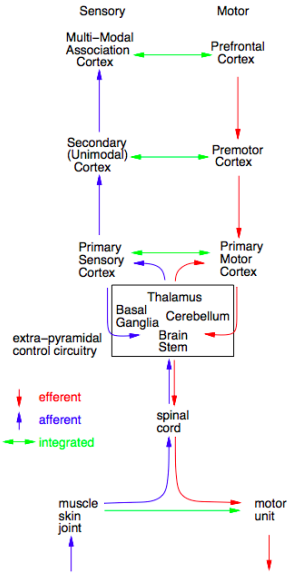
- $\tau_1[2]$, $\tau_2[2]$

mobile base:

- wheel torques $\tau[2]$

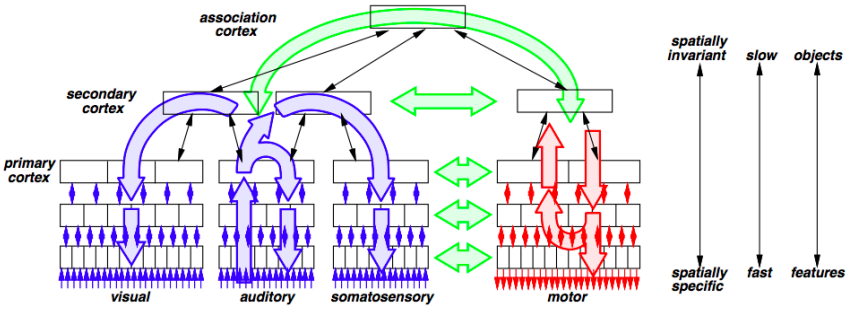
Hierarchical Control



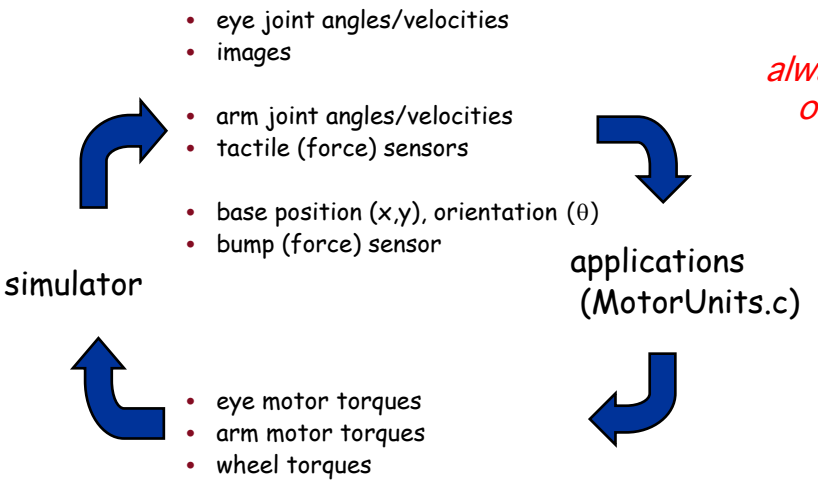


Building Hierarchical Control Code

replicated control structure



Robot Interface: Project #1, #2

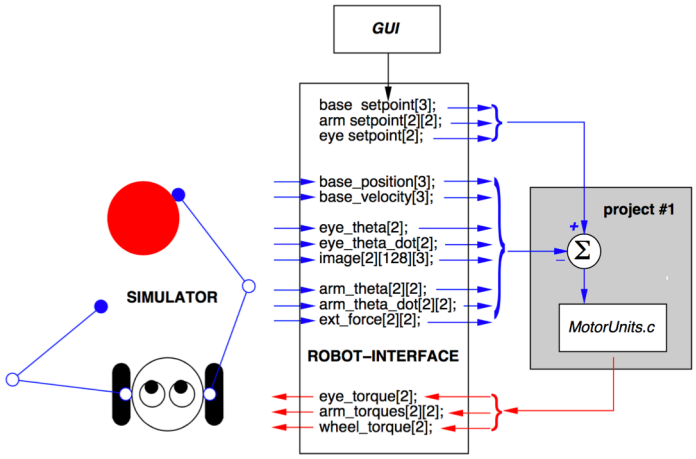


Control Interface - control.h

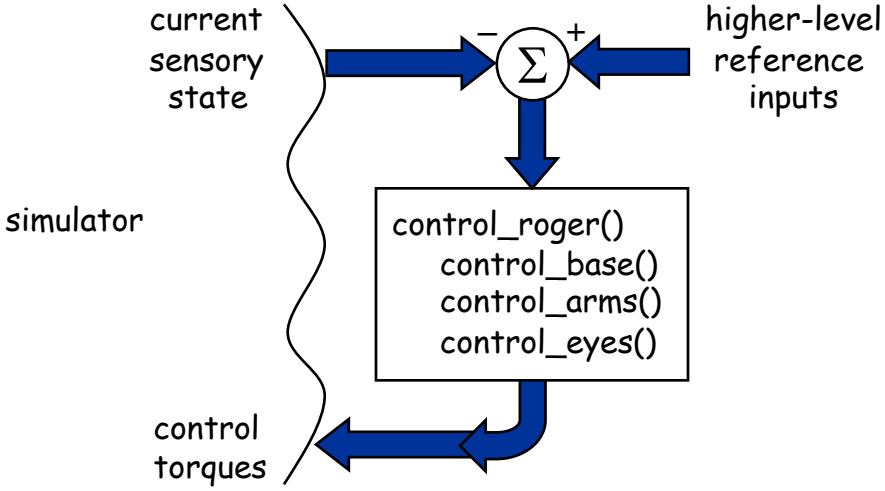
```
typedef struct Robot_interface {
  // SENSORS
  double eye_theta[NEYES];
  double eye_theta_dot[NEYES];
  int image[NEYES][NPIXELS][NPRIMARY_COLORS]; /* rgb */
  double arm_theta[NARMS][NARM_JOINTS];
  double arm_theta_dot[NARMS][NARM_JOINTS];
  double ext_force[NARMS][2];                 /* (fx,fy) force on arm endpoint */
  double base_position[3];                    /* x,y,theta */
  double base_velocity[3];

  // MOTORS
  double eye_torque[NEYES];
  double arm_torque[NARMS][NARM_JOINTS];
  double wheel_torque[NWHEELS];

  // TELEOPERATOR
  int button_event;
  double button_reference[2];

  // CONTROL MODE
  int control_mode;
  int input_mode;
  Map world_map, arm_map[NARMS];

  // REFERENCE VALUE
  double base_setpoint[3];                    /* desired world frame base position (x,y,theta) */
  double arm_setpoint[NARMS][NARM_JOINTS];    /* desired arm joint angles */
  double eyes_setpoint[NEYES];                /* desired eye pan angle  */
} Robot;
```

```
typedef struct _map {
  int occupancy_map[NBINS][NBINS];
  double potential_map[NBINS][NBINS];
  int color_map[NBINS][NBINS];
} Map;
```

Project #1 - Code Structure



MotorUnits.c



Cumulative Project Work

1. motor units
2. Cartesian goals
3. oculomotor behavior
4. visual reconstruction - triangulation
5. “hunting” - integrated behavior
6. ...

<u>options:</u>

1. path planning
2. learning
3. Pong
4....