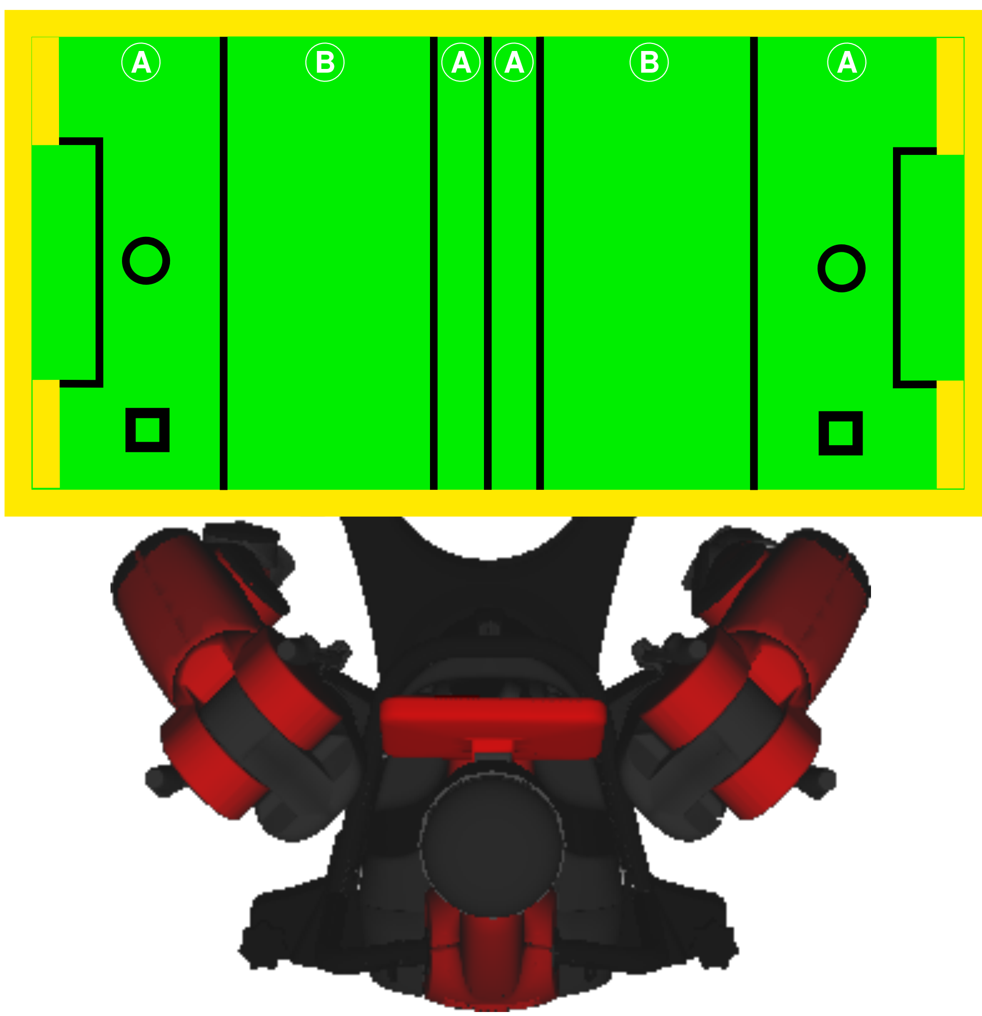
Robotic Manipulation 2015

Final Project (100 points)
Proposal Milestone: 3:35pm, Fri, Dec 4, 2015
Competition 5pm, Monday, Dec 14, 2015
Report due: 5pm, Thursday, Dec 17, 2015
Location: Gates G03/G05 (food will be provided)

v 0.3. Release date:

November 29, 2015

(Updates will be announced on Piazza)



Overview: This assignment should be completed in groups of approximately 2–4. The assignment will involve open-ended head-to-head competition on Baxter between pairs of teams. The challenge is designed to integrate capabilities you have built on Baxter through the semester.

Competition format will be double-elimination with initial match-ups determined at instructor discretion.

Grading will be as follows:

- 85% analysis of your approach (proposal and report)
- 15% based on outcome of competition

OpenCV Tutorial. The final competition will require use of OpenCV. OpenCV has excellent tutorials in Python and C++.

- Python: https://opencv-python-tutroals.readthedocs.org/en/latest/
- C++: http://docs.opencv.org/doc/tutorials/tutorials.html

You will need to use CvBridge in ROS to import images into OpenCV from the appropriate ROS topic. Tutorials in Python and C++ are located here:

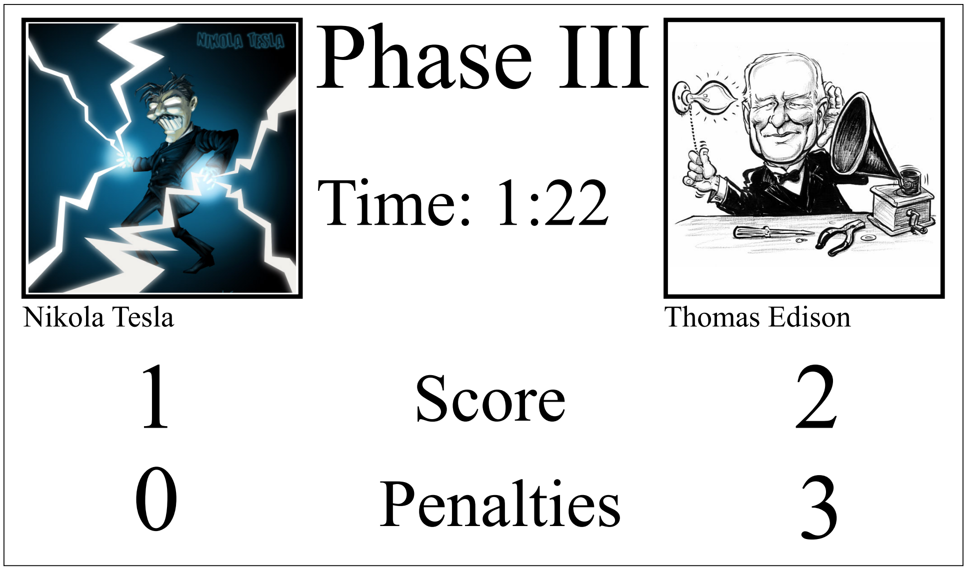
- http://wiki.ros.org/cv_bridge/Tutorials

Competition rules

During a match, each team will be assigned arbitrarily to one of Baxter's two arms. The playing field is layed out symmetrically as shown in the figure above. The field is divided into two sides, each belonging to one team. The various green regions on each side have different rules for manipulation, detailed below. The small ring is designed to contain a ball, whereas the square contains a stack of blocks.

Each team designates a **captain**, who is the only person that may interact with the game during the competition.

The objective is to **score goals** by putting a ball into the opposing goal, while performing defense using the blocks. A secondary goal is to **knock the blocks out of your opponent's Ⓑ region**. The two teams compete autonomously through their own software to achieve these goals. The following image is an example of how the score board, displayed on Baxter's face, will look during the game.



A match consists of three phases. In **phase I**, both teams set up their code and do all necessary calibration. Note that for the final competition, the robot and playing field will be moved to a different room that may have slightly different conditions. Phase I ends when both teams indicate they are ready or when the referee deems an excessive elapsed time (at least 5 minutes). At the end of phase I, each robot arm should be positioned on top of a stack of 2–5 blocks, as in the start of Project 1.

In **phase II**, the teams have up to 2 minutes to arrange as many blocks as they wish from the stack within their own Ⓑ region.

In **phase III**, teams play for 5 minutes. The timer starts immediately after phase II ends. At the start of phase III, the referee places the ball in one team's circle (arbitrarily chosen). Both arms may perform both offense and defense according to the the following rules. The gripper may pick up or interact with the ball in any location on its team's home side. The gripper may also pick up and manipulate blocks that are inside the Ⓑ region. Any block whose center is dropped or bumped into the Ⓐ regions is still in play as an obstacle but cannot be deliberately moved by the robot. A block that is bumped back into the Ⓑ region again becomes fair game for manipulation. Blocks in the green box adjacent to either goal will be removed from play at the referee's discretion.

If the ball comes to rest on one side of the table, the corresponding side's team captain may opt to move the ball by hand back to the circle for the robot to find. If that captain declines to move the ball, then the other captain will have the option to move it to their own circle. The referee may move the ball at his descretion into either circle to maintain liveness of the game (for example, if the ball is in one circle but the robot is not doing anything with it).

If the ball goes out of the bounds, control of the ball returns to the team who touched it least recently.

One **penalty point** will be assessed for each the following events:

- A team member besides the captain interacts with the game (physically or through software).
- The captain touches the ball in motion, touches a block, or touches the robot.
- The robot deliberately manipulates a block in the Ⓐ zone during phase III.
- The ball goes out of bounds.
- Any part of the arm besides the gripper deliberately makes contact with the game (no blocking

with the forearm).

Hardware modifications are permitted within the following parameters. Hardware may be added to the grippers, either singly or bridging across the gap. Any such hardware should be black and should quickly attach to the gripper. There will be a brief opportunity between phases II and III to slide custom hardware onto the gripper. Any custom additions must fit within a box of dimensions 13 cm on a side. No adhesives are permitted – no tape! Rubber bands are permitted and may be used both to affix parts to the robot and to store potential energy.

Scoring: The winner will be determined by the following criteria, evaluated in order. The first non-tie will decide the match.

1. The team with the highest goals minus penalties
2. The team with the highest number of blocks in the Ⓑ region minus penalties
3. Coin toss

Color key:

Field – block containment area	green Ⓑ
Field – non-block area	green Ⓐ
Goal (vertical)	yellow
Golf ball	pink
Blocks	blue
Field markings for humans	black
Other objects to be ignored by the robot	black

Software API

During competition, two teams will execute their code on two different PCs in the Baxter lab. The referee will operate a third PC that runs the `game_server`. This package will be installed on all of the PCs in the lab, and it can be downloaded from
http://notfoo.cs.cornell.edu/repos/game_server

In order to initialize play, each team's code must contact the game server using the service `/game_server/init` of type `/game_server/Init` with the following message type.

```
string            name  # your team name
sensor_msgs/Image logo  # your 300x300 logo
---
string            arm   # which arm you will compete on
                        # arm takes values "left", "right", or "none"
```

That logo will appear on Baxter's display, indicating which side your team will be on for the match. Note that each team must use a unique name string. Re-connecting with the same name will give the same side so that teams may re-initialize during Phase I if necessary.

The following **parameters** will be automatically set for you (do not set them yourself!)

- `/num_blocks` – the number of blocks in each stack at the start
- `/configuration` – set to `/stacked_ascending` (strictly for backwards compatibility)

Game state will be reported periodically to both teams on topic `/game_server/game_state` with the following format.

```
int32    current_phase  # 0, 1, 2, or 3.  0 Indicates game is not running.
duration time_remaining # seconds
int32[]  score          # Use the following as indices to get team score
int32    LEFT=0
int32    RIGHT=1
```

Teams are responsible for reading the game state and acting according to the rules of phaess I–III.

Deliverables

Besides the competition itself, you will be graded on a written report in which you analyze your implementation from a theoretical perspective. As a milestone, you must submit a short "proposal" by the time of class on Dec 4.

Please state the division of labor in your report. It should be clear who is responsible for what aspects of implementation and analysis. "We all worked together" is not an acceptable division of labor, especially with larger teams.

The report must show mastery of some theoretical concepts from the course in several areas. The areas are

- kinematics,
- grasping,
- motion planning,
- control,
- vision/visual servoing,
- dynamics, and
- force control.

You may build on mathematical analysis from the textbook or go beyond or consider alternate techniques. Teams containing 4000-level students must show mastery in at least 2 areas from the list, and teams of 5000-level students must show mastery in three areas. Reports from mixed-level teams will be evaluated differently for each student according to their level. One page per area is reasonable, but more may be necessary for some topics. Additional areas of analysis beyond those required will be considered extra credit, including outside areas not covered in the course. Some applicable outside areas that would be elligible for extra credit include:

- machine learning,
- localization,
- mapping,
- RGB+D sensing, and
- stereo vision.

Grading criteria for each area:

1. Identify the problem you are solving by posing a substantive, specific "how" question.
2. Introduce equations or theorems relevent to the analysis the topic of the question.
3. Identify approaches, techniques, or algorithms appropriate to solving the question.
4. Apply such approaches, techniques, and algorithms to solve the problem.
5. Describe how these ideas were implemented in code.

For example, for the topic of kinematics, one might pose the following question:

$\boxed{\text{How can we choose coordinate frames for the links of a robot arm to consistently solve for the joint angles given the position and orientation of an object to grasp?}}$

The solution of the problem might involve a description of the the Denavit-Hartenberg convention, including the equation of the A matrices, as well as a discussion of Baxter's forward and inverse kinematics as they apply to the problem. The analytical approach to IK is not the only choice here; a numerical method would be equally valid.

The proposal due on Dec 4 should state the areas and questions you plan to address in your final report. One paragraph per topic is enough. The proposal should be submitted by email to the instructor, rak@cs.cornell.edu.