METR 4202 -- Advanced Controls & Robotics

Lab 2: Sensing & Perception --
Robotics. Close at Hand

Ah, but a man's reach should exceed his grasp,
Or what's a heaven for?
Robert Browning

Objective

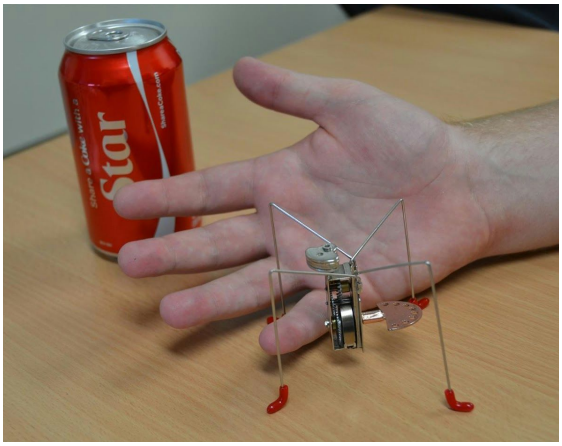
This laboratory explores sensing hands-on. It will investigate sensor interfacing and processing as well as the nature of state-space observers in the context of estimating and tracking the location and pose of a common, handy object – a hand!

This is explored with the help of a RGB+D camera (or an equivalent sensor of the team's choosing). In this laboratory, teams will:

- **calibrate** the RGB+D camera
- **locate** the camera/sensor relative to a base frame
- **segment** the hand(s) in the scene
- **estimate** the location of hands and/or objects in the scene
- **track** the location of moving hands

The laboratory is broken into two sections – core tasks ("the grasp") and challenge tasks ("the reach"). The former are required of all teams, whereas the latter allow teams to obtain (high) distinctions. Performance is seen as a cascade. To borrow the vernacular is measured "applause". Basic performance ("clap") shows standard understanding of the core concepts, whereas exceptional performance ("ovation") are those that adeptly and automatically exploit all structure at hand.

What that robotics is now close at hand!



Workspace

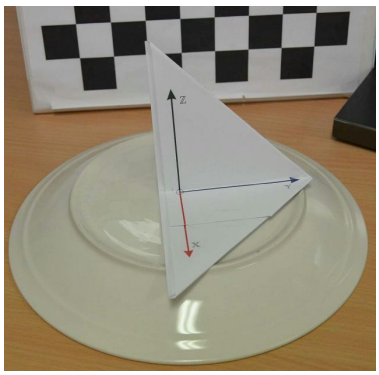
The workspace is similar to Lab 1 with extended height. The base plane is 320mm × 192mm (a standard Lego Mat). The height is 500 mm (or about five times higher). Especially for operation at more advanced levels, it may include clutter in the form of lego pieces, marbles, cups, coins, soda beverage cans, small chocolates, and other small random items that seem to come to hand (and in homage of previous years' METR4202 course projects).

Scene Structure

The workspace will be defined with varying levels of structure and clutter, with lower performance standards having more structure. This is outlined as follows:

Item	Basic Level	Advanced Level
Background	Monocolor non-white paper (teams may remove it)	(Anything)
Clutter	No	Yes
Colour Calibration Target	Optional	No (except for core task 1)
Central frame placement	Static (Optionally team-placed)	Dynamic (Random)
Calibration Pattern	Provided	Optional

Central Frame



The scene will consist of a central frame. It is defined as being right handed with the *x-axis* defined as being oriented towards the camera (parallel to the table surface) and the the *z-axis* upwards (presumably in line with the gravity normal) and the *y-axis* being orthogonal to these axes (see illustration at left)

The central frame might not be on the table surface (i.e., it may be elevated up to 100 mm and at orientation). Its location may be static (meaning that it doesn't move) or dynamic (meaning that the tutors may move the origin between attempts). The origin location is determined by using a random point generator (similar to the one used in Lab 1). In static cases, the origin is nominally the first point of this set (unless teams wish to place the frame manually). In dynamic cases, the tutors may select (and switch between) any of the six points generated by set generator.

Calibration

In order to obtain a useful measurement, the Kinect needs to be calibrated first. Let us begin with the camera. As noted in class, the calibration parameters to connect raw pixels images to 3D measurements are:

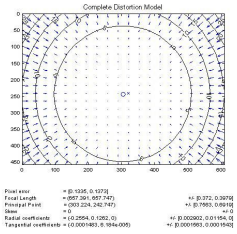
- focal length at the center ($\mathbf{f}_c$)
- principal point offsets from the center ($\mathbf{c}_c$)
- lens skew and distortion ($\boldsymbol{\alpha}_c$)
- Orientation ($\mathbf{R}_c$) and Position ($\mathbf{p}_c$) of the camera

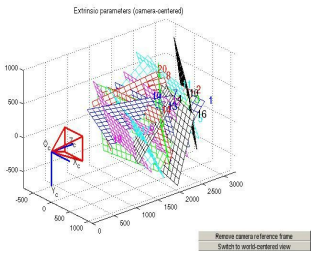
There may be a misalignment of the *x*-axis of the image coordinate system and the base line. However, this can be ignored if the depth coordinate is defined parallel with the image coordinate system instead of the baseline, but this makes later coordination of sensor data to motion complex.

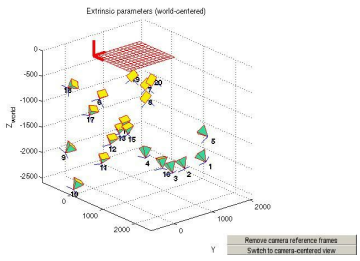
Planar target camera calibration (Zhang's method) uses multiple viewpoints of a calibration target with calibration determined by correlating known points on the target with observed points by the camera. A common target (used by the calibration toolbox and OpenCV) is a black and white planar checkerboard as the corner points have high contrast allowing for (subpixel) precision even with noisy camera and target printing. In this laboratory, while camera placement is an option, the view is fixed. Thus, the camera has to be calibrated before; or, the target needs to have motion (e.g., be attached to a soft spring, such as a string holding the target, or even via a mobile); or, there has to be be multiple viewpoints (or calibration targets) in the scene (e.g., a calibration cube).

Calibration will provide the intrinsic (perspective camera model parameters) and extrinsics (camera pose relative to the target). However, one might ask if this is necessary, as one way to frame the problem is to put the camera well overhead, thus making the scene essentially para-perspective. While this could even make calibration a direct linear (scale) operation, it comes at the cost of less occlusion robustness and no direct means for radial distortion correction. An example calibration (from the Bouguet Calibration Toolbox) is shown below:

```
Calibration results after optimization (with uncertainties):
Focal Length:          fc = [ 657.39071   657.74678 ] ± [ 0.37195   0.39793 ]
Principal point:       cc = [ 303.22367   242.74729 ] ± [ 0.75632   0.69189 ]
Skew:             alpha_c = [ 0.00000 ] ± [ 0.00000  ]  …
Distortion:            kc = [ -0.25541   0.12617   -0.00015   0.00006  0.00000 ] ±
  [ 0.00290   0.01154   0.00016   0.00015  0.00000 ]
Pixel error:          err = [ 0.13355   0.13727 ]
```







The basic level of calibration would be calibrate this in advance before the laboratory starts, whereas the more advanced level would be to calibrate camera online during the laboratory.

Core Tasks: Give Us a Hand

The core tasks in this laboratory are focused around basic concepts around the kinematics and geometry of vision/sensing. They include:

(1) Calibration (camera geometry and/or colour)
(2) Sensor Localization/Placement (sensor pose may vary)
(3) Object detection (is there a pen(s) or a (single) hand in the scene)?
(4) Object position determination (particularly in the presence of noise and clutter)

Notes:

- The number of "claps" next to each level is indicative, "up to" and **not** absolute.
- In general, basic levels correspond to basic level environments, whereas ovation level corresponds to advanced levels of performance.
- Except for the eigen level, the frame may not be placed on the object of interest.

Details:

1. Camera Calibration

Teams should **fully** calibrate their camera (i.e., intrinsics and extrinsics). Teams are allowed to choose their pattern, but an A5 or larger **30 mm** square checkerboard pattern (the metric version of the checkerboard pattern provide in the Camera Calibration toolbox and the same format as in Tutorial 6) will be available.

- **Basic Level** (1-2 claps): Calibrate the camera in a "Basic Level" environment. The teams may move the target themselves. While automatic operation is recommended, semi-manual operation (1 clap) is allowed. Automatic calibration (2 claps) of the camera may allow manual pattern motion (i.e., "calibration dancing").
- **Applause Level** (4 claps): Fully automatic calibration of the camera in an "Advanced Level" environment. Manual pattern motion is also allowed.
- **Ovation Level** (6 claps): "Applause Level Camera Calibration," plus the calibration should be done without any manual pattern motion, meaning that the multiple calibration pattern views must be done automatically (passive motion (e.g., spring, mobile, etc.) is allowed). (*hint*: what is the minimum number of views and the minimum number of points that need to be tracked in these views necessary for calibration?)

Colour Calibration

In addition (as in optionally), teams may also color calibrate their cameras. Take an image of a Gretag Macbeth Colorchecker target find the returned value of the **Neutral 3.5 (#555555 - chart #23)** and **Light Skin (#c29682 - chart #2)** colour targets in RGB, HSV and L*a*b color space (hint: see the applycform and rgb2lab functions).

- **Basic Level** (½-1 clap): Manually or automatically Identify the chart in a "Basic Level" environment and determine the RGB and HSV values for the chart values #12 and #21.
- **Applause Level** (2 clap): Automatically identify the chart in an "Advanced Level" environment and determine the RGB, HSV, and Lab values for at least chart values #12 and #21. (*hint:* what is an appropriate illuminant model for the room?)

2. Sensor Localization

The system should locate the camera (in metric coordinates) relative to the central frame. Teams may provide a central frame target of their own choosing (i.e., using any given design or their own as long as the frame chosen does not obstruct operation/marking). If teams chose to place the frame manually (as part of a "Basic Level" scene) they will suffer a 1 clap penalty.

- **Eignen Level** (0 Claps): The central frame is placed at the camera origin. (this is inclusive of the manual frame placement penalty)
- **Basic Level** (2 Claps): The central frame is placed in the workspace and if a target is used, then the target is visible by the camera. The estimated location is within 10 cm (total straight line distance) of the actual value (as surveyed by the tutors)
- **Applause Level** (3 Claps): The central frame is placed by the tutors to the left of the plate. The estimated location is within 10 cm of the actual value. Pose estimates are attempted.
- **Ovation Level** (5 Claps): Now in an Advanced environment, the system locate the camera automatically to an arbitrary location. The system also determines pose. The estimate is within 5 cm. The pose estimate is within 15°.

3. Object Detection

The system should be able to detect if there is are pen(s), a hand or nothing else in the scene.

- **Basic Level** (2 Claps): The system is able to detect that a single hand is in the scene or not. Operation is in a basic environment. The system may request in advance to have the hand oriented palm-up or palm-down. It need not be able to detect if pens are in the scene. Teams may request that the hand is placed on the table or approximately at a preset height of the table (e.g., via a guide/rest for the arm).
- **Applause Level** (3 Claps): The system should detect the presence of a hand automatically at palm-up or palm-down pose and at any arbitrary yaw angle. In addition to the system is able to detect that a pen or pens (of the type used in Lab 1) are in the scene. Operation is in a basic environment.
- **Ovation Level** (4-5 Claps): Now in an Advanced environment, the system is able to automatically detect the presence of a hand in any arbitrary orientation (yaw). It should be able to detect pen(s) even in the presence of clutter. Optionally, the system should try to be able to place a bounding box around the hand and display this as well.

4. Object Position

The system should be able to return an estimate of the (ideally metric) location of a hand and/or pen(s) on the table surface (i.e., the objects of interest in stage (3))

- **Basic Level** (1-2 Claps): The system is able to return a 2D estimate of the hand's location in pixels (1 Clap) or in metric (2 claps) relative to an arbitrarily placed base frame. Operation is in a basic environment.
- **Applause Level** (2-3 Claps): The system should return the 3D metric position of the hand relative to the based frame from stage 2. Operation may be in a basic environment (2 claps) or an Advanced Level environment.
- **Ovation Level** (4-5 Claps): Now in an Advanced environment, the system is able to return the automatically the metric location of the hand and additionally estimate the position of the pen(s) on the table. Optionally, the system should return the orientation (yaw) of the hand as well.

Challenges: Let's Try a Hand at It!

It is expected that the challenges are done in an advanced environment. A reasonable attempt is one where the solution has a likelihood of producing a correct result. Whereas robustly successful solution is one that yields correct results reliably under varying conditions.

(1) **All Hands on Deck!**
- The system should be able to detect (and ideally locate) Multiple hands

(2) ***À Quatre Mains* (Piano four hands)**
- The system should track Multiple (up to four) hands in motion (approximately) along a line. Optionally there may be a printout of a piano keyboard for the hands to follow.

(3) **Hands Up! (aka: Hand me Down!)**
- The system can cope with hand pose variations along all degrees of freedom in the wrist (i.e., yaw, pitch and roll [or in terms of anatomy: extension/flexion, radial/ulnar deviation and circumduction]).
  (*hint: the wrist has a limited range of motion*)

(4) **Handwriting**
- Find the number of hands holding a writing instrument (such as the pen from Lab 1)

(5) **Hand-righting**
- Detect (and ideally locate) the number of "right" hands in the scene

(6) **A Bird in the Hand**
- Track the hand even with large occlusions (such as holding a bird)

(7) **Cup in Hand**
- Determine the size of a cup (small, medium, or large) held in a hand.

(8) **Hand in Cup**
- Determine the volume of a hand (up to the wrist as demarcated by a wristwatch). The solution may be interactive and may request the user to follow a prescribed set of motions. The volume is validated by having the hand immersed in cup (beaker) of water and measuring the fluid displaced.

(9) **Put one's finger on it**
- Determine the angle (relative to the base frame) that a person is pointing at.

(10) **The Dark Side!**
- Remove the shadow cast by a hand. The solution just has to display an image with the illumination variation created by the hand's shadow removed. It may or may not remove the shadows created by other objects in scene.

(11) **I have to hand it to you…**
- Teams (in consultation with teaching staff) may propose their own.

Marking: Assessment Criteria for Overall Lab Mark

Despite the hand-wringing, let us consider the process of handing over grades:

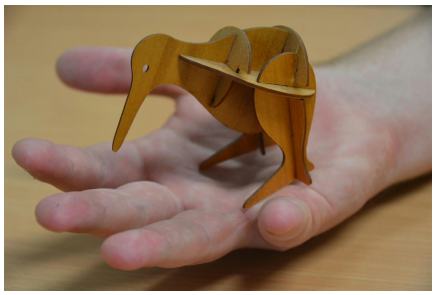
In short for the following grade levels:

- 1-3: Teams attempt and are somewhat successful at 1, 2 or 3 of the "core tasks".
- 4: Teams attempt some "core tasks" and are marginally successful at all core tasks
- 5: Teams attempt most "core tasks" in an advanced environment and are robustly successful at all core tasks even, including the presence of noise/clutter/etc.

- 6: Teams attempt at least 2 and are successful at 1 of the Challenge Tasks
- 7: Teams attempt at least 3 and are robustly successful at 2 of the Challenge Tasks.

As a rough guide that mapping between claps and the grades is:

Grade	Applause Level	Description
2 (20-45)	2 claps	At least one task performed. For example, you are able to calibrate the Kinect in both color and metric space.
3 (45-50)	3-6 claps	Very substandard performance, For example, you are able to detect the presence of objects in one basic scene.
4 (50-65)	7-10 claps	**Basic level operation.** For example, you are able to detect the location of at least one target object in some of the scenes with satisfactory accuracy.
5 (65-75)	11-15+ claps	**Intermediate operation level.** For example, you are able to detect the location of a hand in the scene with good accuracy.
6 (75-85)	12+ claps + ≥1 Successful Challenge	**Very good intermediate to Advanced Level performance.** For example, you are able to detect the location of some target objects in each scene with great accuracy.
7 (85-100+)	15+ claps + ≥2 Successful Challenges	**Excellent performance**. Most of the tasks are attempted well. Teams are able to robustly detect the locations of objects in complex scenes with superb accuracy.



Teams and Groups

The project will be conducted in **teams of four** (up to five maximum) -- preferably taken from within your group from Laboratory 1. You may also choose an individual from another group (or laboratory session) as long as you understand that you may not be able to work together for the final project that will draw upon work completed in this one. A requirement to pass this Laboratory is that you must be in a team and that team must be registered in Platypus[2] by September 5, 2015.

Other Programming Systems and Cameras

Teams may elect to use programming languages and systems other than Matlab, such as Visual C or Python (i.e., the class is language / system neutral). In particular, teams may choose to use OpenCV.

Teams may choose to operate the Kinect's RGB camera in high resolution mode (i.e., they may use the 1280x1024 mode also provided by the MS Kinect SDK as compared to the 640x480 default mode provided by the OpenNI SDK). Similarly, teams may use another (web)camera on the proviso that the camera is autonomous (i.e., it can take pictures without manual intervention) and that the maximum resolution is set to (or automatically down-sampled to) 1280x1024. (n.b., the allure of high-resolution can be a trap in vision and signal processing applications as this comes with higher data processing requirements and often comes with more noise).

In both cases, however, the only supported programming system and hardware are MATLAB the the Kinect camera (using OpenNI/Primesense drivers).

External Sites/Programs

Some external programs and site that might help with the process are:

- **CONDENSATION** -- (M. Isard and A. Blake. "Condensation—conditional density propagation for visual tracking." *Int. J. of Computer Vision* **29(1):5-28, 1998**)
- **CLAMS -- Calibrating, localizing, and mapping, simultaneously**

Due Date

The laboratory must be completed by **Thursday, September 24, 2015**. The code should submitted online via a source version control system (e.g., EAIT GIT, GitHub, GitLab, Bitbucket) by 11:59pm September 26, 2015. A short team report should by 11:59pm on September 26, 2015 via the Platypus[2] submission system. Early submission is encouraged.

Demonstration

As with laboratory one, the system will need to be demonstrated. As with Laboratory 1, there will be signup times on Thursday September 24 and Friday evening September 25 (i.e., during teaching week 9). During the demonstration period, teams may choose to demonstrate the focus area tasks in **any order** they choose. For each of the tasks, teams may repeat a task once if they choose; however, the team receives the value from either not both (i.e., repeat task demonstrations do not add).

Given the number of teams, the demonstration times (of 15 minutes total including setup, leaving 5 minutes for discussion) will be strictly enforced. It is recommended that teams come 15-30 minutes in advance of their demonstration appointment. It is also recommended that teams practise their demonstrations as time limits will be enforced even if teams have not been able to demonstrate their solutions to the five tasks (i.e., teams will receive grades not on the solutions they demonstrate not the solutions they might have, but did not deliver).

Deliverables & Submissions

It is also required that all code be submitted (via version control system). As the laboratory is language and camera neutral, teams that do not use Matlab should provide very clear documentation and README files with their submissions.

Output has to be in a human readable format. An example structure might be:

1. **Camera Calibration:** given a sequence of RGB+D data containing checkerboard patterns from an uncalibrated Kinect (or RGB for a camera), return the intrinsic and extrinsic constants of the Kinect. The returned intrinsic properties should be in a structure like following, the square brackets indicate the expected size of each property. Note that the extrinsics has three dimensions; the third dimension has a transformation matrix for each RGB calibration input image:
   intrinsics.fc = [1x2], intrinsics.cc = [1x2], intrinsics.alpha_c = [1x1],
   intrinsics.kc = [1x5], intrinsics.err = [1x2], extrinsics.transformation_matrices = [4x4xN]

2. **Detection:** given an image from the Kinect/camera, it must return the hand location(s) in a standard format.

3. **Localisation and Mapping**: given an image from the Kinect/camera, it must return the location of the camera relative and/or coin(s) (depending on the mode) relative to a central coordinate frame in metric coordinates (mm) with orientations in (degrees). There should be a row for each coin located. There should be 3-6 columns to represent the coin's pose: [*xPosition, yPosition, zPosition, Roll, Pitch, Yaw*]. Orientation may be in Quaternions, etc. if specified in the header or function documentation.

Judges

The course coordinator, lecturers and tutors will act as judges. The course coordinator will act as chief judge. All decisions made by the judges will be final, and no correspondence will be entered into. Contestants may approach the organiser about possible designs that may be questionable under the rules listed above. Any queries will be treated with the utmost confidentiality and will not be divulged.

Custom Levels

As custom level ideas are sent in and approved, they will be posted here for the benefit of other teams. Some approved custom advanced level and extra credit ideas are:

- Open Source Code -- The entire code base is shared on a public, open-source repository (e.g., GitHub) = +1 clap
- Contribute to Libfreenect2 -- Develop and share drivers/bindings = +1 clap

Caveats

Some general "reasonable person" rules apply to the code and its execution:

- It is expected that teams will use source/version control
- Codes with fixed (predetermined) estimates are not valid (even if the value is correct).
- The use of the Matlab Camera Calibrator App may be used. However, teams will still be responsible for being able to explain how the calibration process works, particularly intrinsics, extrinsics, the minimum number of frames and points required (see also § 2.1.5 (pp. 45-49) of Szeliski, *Computer Vision: Algorithms and Applications*, 2010).
- Internet access may or may not be present -- the code should assume that it will not have Internet access during execution and thus operate in a self-contained manner. This proviso excludes UQ license servers that may be needed by the program (e.g., Matlab). A "Mechanical Turk" or "phone home" solution is explicitly disallowed.
- Memory space may or may not be cleared between challenges and submissions -- The memory space might be cleared before each function. Thus, if your routines rely on parameters to be exchanged, it should do so by writing to a file. Similarly, if certain variables names (e.g., counters) are used between functions, then be sure to initialize them correctly.
- Each team's submitted functions will be run in their own directory -- Reading other teams' files or memory is disallowed.
- All source code(s) may be assessed -- Thus, it is requested that it is commented. If custom precompiled codes are used (e.g., mex files), the source code and compilation instructions (e.g., makefiles) should also be submitted.
- Computational and memory resources -- the functions should be able to operate reasonably on a "standard" Laptop/Workstation class computer (such as the UQ EAIT PC Workstations). Judges may terminate execution after 2 minutes.

METR 4202: A Handy Subject!