Objective

Use a depth camera and a turntable to create 3D representations of objects. Use the depth camera to generate point cloud data and a turntable (with position feedback) to spin the object and map all sides/features. The utility of this is that the full representation of the object could then be used for collision detection, grasp planning, path planning, etc, whereas with only a single-side representation, large clearances may be required to ensure no collisions are made with the unknown sides of the object, a grasp might not be possible, etc. ROS will be used as the framework, and the majority of the required code will be written in C++. Effort to tie in concepts from concurrent computer vision and sensor network courses will be made wherever appropriate, as well as relevant past course work (ME495, EECS469).

Process, Milestones, Timeline

1.	Prework: gather required info, order required hardware (turntable with feedback), determine object(s) to use, workspace setup, assess feasibility with Jarvis / prune project scope, etc	1 week
2.	Depth camera correctly differentiates between object of interest and turntable/background	2 weeks
3.	Determine and store 3D point representation of visible surfaces (static, no turntable rotation involved yet)	2 weeks
4.	Turntable rotation added, dynamic surface addition working and capable of 3D point representation of more than just the visible surface	4 weeks
5.	Markers added to the 3D representation to ‘see’ occluded sides/features of the object as it is spun, saved to object file as they are added	1 week
6.	Implement any stretch goals, if time allows	
7.	Project demonstration, documentation, wrap-up	1 week

Stretch Goals

- write an algorithm to determine the optimal gripping configuration (required pose of the gripper relative to the object to best pick it up)
- *even stretchier:* then use Baxter and the turntable to execute the grasp
- remove turntable feedback and calculate the angle using the RGBD information
- *even stretchier:* make this work like SLAM (once features are connected, updates the positions based on most likely case - could potentially modify existing SLAM algorithm)
- Write an algorithm to simplify the point cloud representation of the object. For example, remove unnecessary/extra points from planar surfaces, edges, etc.
- incorporate depth camera measurement uncertainty to provide pseudosurfaces representing 95% confidence intervals?

Questions / Unknowns

- is it better to do average of points (i.e. spin the turntable 3 times and map the features from the average of the 3 spins)? Would this require a Bayesian filter?
- Which camera would be best for this purpose? Xtion? Kinect? Other?

- Any chance there will be resolution issues? What is the minimum distance between camera and object that the camera needs to detect?
- What is tolerance on depth measurements?
- Will distortion be a problem? Is it possible to calibrate to reduce this distortion?