The Kinect has several disadvantages that had to be overcome, largely due to the closed-source nature of the skeleton-tracking software. The Kinect has a limited field of view (57 degrees). The Kinect was designed to track users from a fixed vantage point. The Kinect is sensitive to vibration. The Kinect does not work well in direct sunlight.

The Kinect is accessed through an open-source API called OpenNI (Open Natural Interraction). However, the actual skeleton tracking is done by a closed-source binary (NITE, made by PrimeSense.) NITE provides few options for configuration, so at times software workarounds had to be employed.

# CALIBRATION

By default, whenever OpenNI detects a new user in its field of view it requires the user to stand in a calibration pose [PICTURE] to take an accurate measure of the user's limbs. This calibration step takes several seconds and requires the target to be still.

When the Kinect is on a moving base, occasionally the target will be lost due to relative motion or jolts (as discussed later). Upon target reacquisition, recalibration would frequently be required. Recalibration woudl require both Harlie and the target to come to a halt, which is unacceptable given the goal of smoothly following the target. Luckily, through somewhat of a hack, OpenNI can be instructed to save the calibration of the first detected user, and for all subsequent users skip calibration and just use the saved calibration.

Skipping calibration comes at a cost. A positive aspect of the calibration step is that the distinctive pose required for calibration eliminates the problem of the robot following the wrong user. Without calibration, the robot no longer has an easy way of telling which user to follow. Additionally, when on a moving base, the Kinect tends to detect chairs as users. These chairs would never pass the calibration step, although without calibration they appear as spurious readings.

Another issue with OpenNI is that by default, the software tracks the entire human body (head, arms, torso, and legs). Harlie's Kinect is mounted in such a way that it cannot see legs close up (INSERT MECHANICAL DRAWING OF KINECT'S FOV). Luckily, the NITE software can be commanded to ignore legs and just track the target's upper torso. However, there is a tradeoff, because without legs, the tracking software loses an important characteristic that can discriminate people from inanimate objects.

# DISCRIMINTATION BETWEEN USERS

The Kinect has no built-in facilities for discriminating between users. No information is stored about a user other than calibration measurements representing limb lengths. By default, the Kinect likely relies on continuity between frames to maintain a lock.

If a user exits the scene, there is no guarantee that when the user is re-detected that user will be assigned the same ID. The same is true if a target is momentarilly lost. If the target is moving and the Kinect encounters a sudden jolt, the target will likely be lost and it may be reacquired with a new ID.

My solution was to use the Kinect as one of several inputs to a Kalman filter that tracks the overall hypothesized location of a person (to be discussed in a later section.)

# LIMITED FIELD OF VIEW

The Kinect has a field of view of 57 degrees. While this is sufficient for tracking a target with limited freedom from a fixed vantage point, it shows weaknesses for moving targets.

When using the Kinect as the sole source of observation, the robot must constantly face the user (within +- 30 degrees) or lose a target lock. This puts severe constraints on its ability to maneuver and plan paths without losing the target.

Even following a target down a straignt hall one can run into problems. If an obstacle appears between the user and the robot, the robot will have to navigate around the obstacle. (INSERT IMAGE HERE OF ROBOT AVOIDING OBSTACLE, INCLUDING TANGENT ANGLE DEVIATION FROM STRAIGHT PATH) Likely, as part of the avoidance the user will leave the Kinect's field of view, leading to a target loss. When the robot once again faces the user, it will have to re-find the user.

The situation becomes worse when the user takes a twisting path, especially doubling back behind the robot. In tight spaces such as hallways, the user will necessarily come close to Harlie when moving behind it. Because the Kinect cannot track targets closer than two feet away, this frequently leads to target loss. Additionally, the robot can get stuck against a wall with no hope of recovery.

# MOVING BASE PROBLEM

The Kinect was designed to track skeletons from a fixed vantage point (in front of a television.)

The Kinect was not meant to be mounted on a moving base. I performed some tests to characterize how well the Kinect can track moving targets from a moving vantage point such as Harlie's back. First, the robot was rotated back and forth through 1 radian of angle (approximately the Kinect's FOV) with a sinusoidal velocity profile. Second, step commands were given in angular velocity and the affect on tracking was observed. Target acquisition and tracking was tested up to 0.8 radians/second in both cases.

When the Kinect is still, its performance was best. The Kinect can detect users rapidly moving through the scene, and it can easily deal with partial occlusion. The Kinect only lost a lock when a target moves very quickly, or exits and reenters the scene. The Kinect can be confused if two users come close together, not being able to tell users apart by means other than their spatial positions.

As expected, the Kinect's performance gradually degrades with the speed of rotation. With a peak or step inputs below 0.3 radians/sec (17 degrees/sec), the performance is almost identical to the case of standing still. The Kinect is still able to robustly track targets through its field of view without many drops.

As the maximum speed increases, performance starts to degrade. Around 0.5 radians/sec, the target will occasionally occasionally be dropped. Usually the Kinect will reacquire it right away, resulting as a flickering effect as the Kinect struggles to keep a lock. The incidence of flickering increases with speed, and well as the chance that the Kinect will lose a target and not reestablish it. Target loss was especially noticable when the Kinect is jolted or given a sharp step command in velocity.

The quality of target tracking is also dependent on the environment. The Kinect is able to track closer users more reliably, likely because they have a greater number of constituent pixels. Occluded users are more dificult to track, especially if the occluding object is close by [INSERT PICTURE OF CHAIR BEING INTERPRETED AS BODY PART].

All of these effects are dependent on speed. Even so, at the maximum tested speed of 0.8 radian/second, the Kinect performed remarkably well given that it is operating outside of its design parameters. Although it frequently loses a lock, it is usually able to reacquire.

Nevertheless, the problems are significant enough to warrant addressing.

Relative motion is a more significant problem than absolute motion. If the target keeps pace with the robot's rotation to remain in the center of the Kinect's FOV, tracking is very accurate.

Targets further away from the Kinect

I hypothesize that the Kinect uses background subtraction and tracks users frame-to-frame. Tracking is a more difficult problem when the base is moving. If the