EE149 iRobot Localization Project:

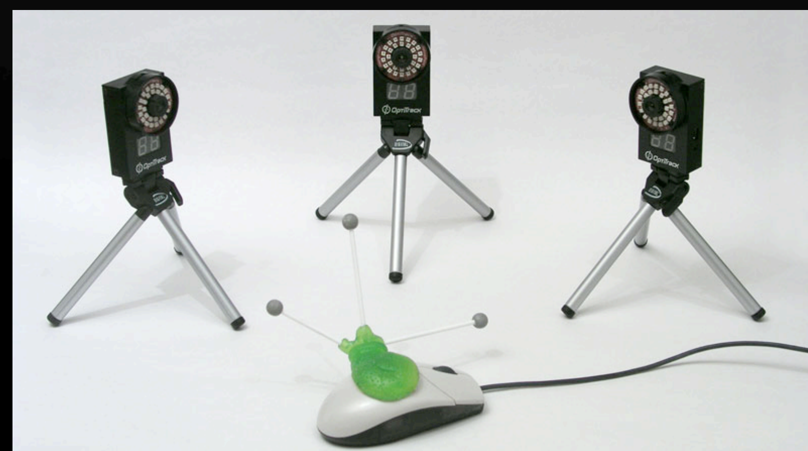
- Motion Tracking system that localizes the iRobots and streams Data to them.

Tracking Systems:

- **Optical Tracking:**

  Triangulate the 3D position of sensors

  - Passive Marker (Retroreflective Markers) (OptiTrack System)



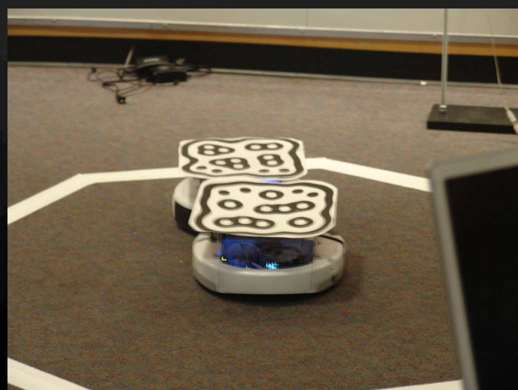
Tracking Systems:

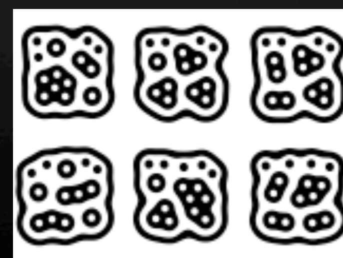
- Active Marker (LED) (Phase Space)

- Markerless (Using algorithms to analyze optical input and identify the object “human body”)

Tracking Systems:

- **Vision-Based Systems:** ReacTivision





- **Inertial Systems:**
  Usually use gyroscopes to measure rotational rates (Wii Remote)

OptiTrack System:

Tracking Tools Package:

5 OptiTrack R2 Cameras (3.5mm lenses) + Reflective markers

Provides 1-Marker and 3-Marker Calibration.

Calibration:

Tracking tools provides both 3-Marker Calibration, and 1-Marker Calibration, and a built-in Calibration engine.

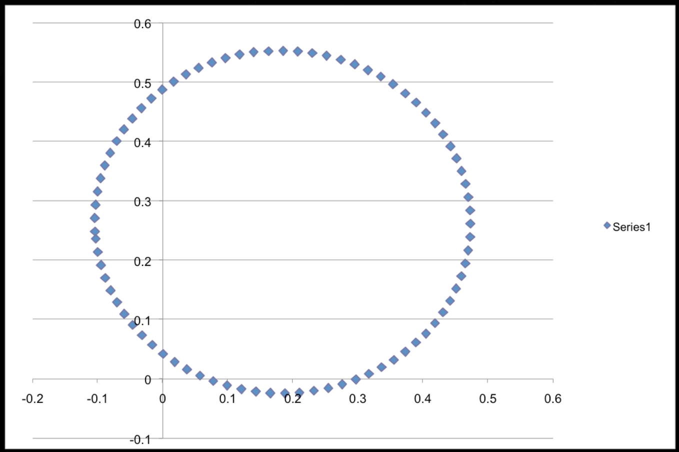
When enough data is collected the engine tells you to stop wanding, and does an initial solution estimation and minimize the 3-D triangulation error by globally optimizing both camera's extrinsic position and orientation as well as its intrinsic focal and lens distortion characteristics.

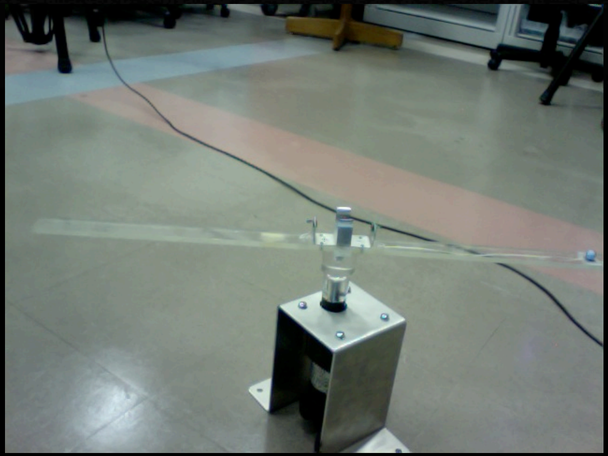
Data Collection:

We also collected data using the circular motor, to calculate the optitrack's system error: rms error: 0.0006549m

Avg. [(R – Avg.[R])^2] = 0.0006549m

R = Avg. [sqrt [(x-Avgx)^2 + (y-Avgy)^2]]



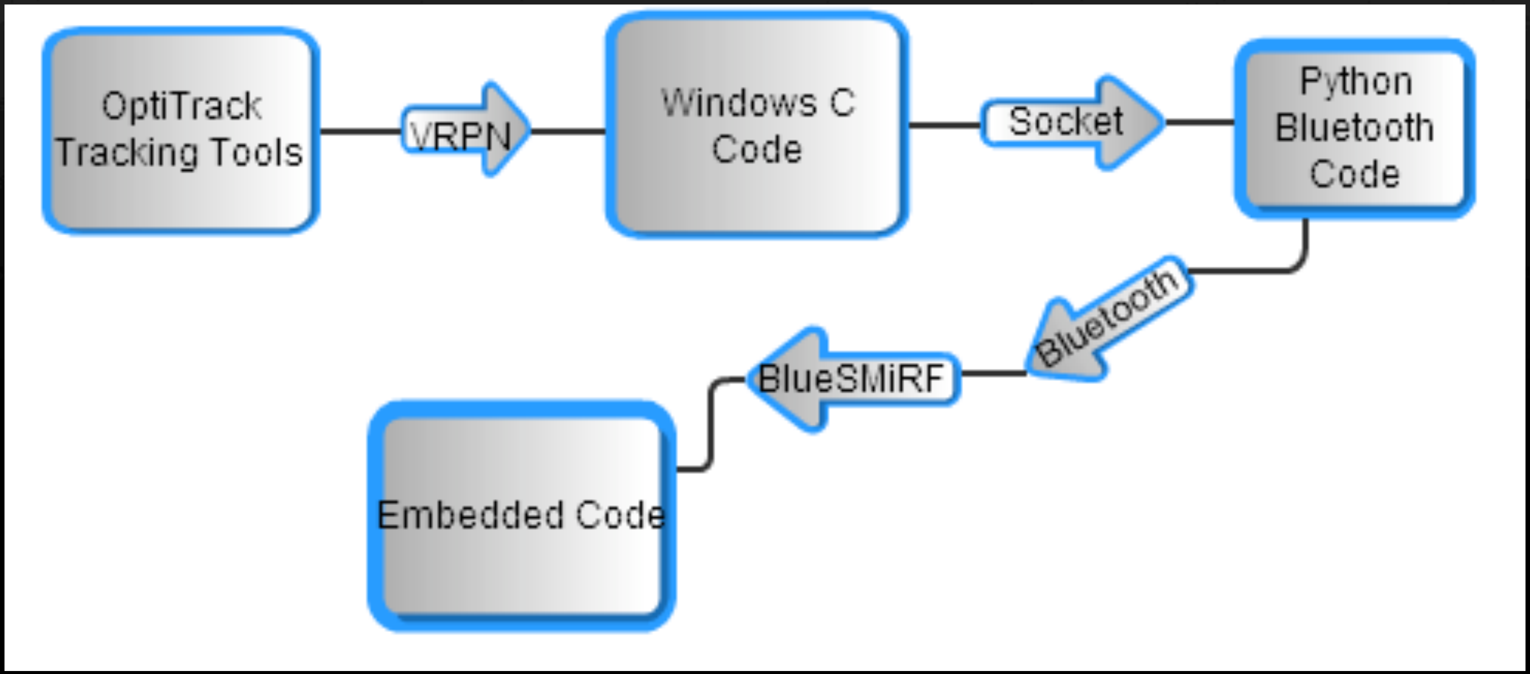


Data Process:

- Export to CSV format
- Real Time Streaming:

NatNet Engine:	Streams Rigid bodies + markers UDP multicast Port 1510
VRPN Engine: **Virtual Reality Peripheral Network**	Streams Rigid bodies TCP+UDP selectable port, default port 3883
Trackd Engine:	Streams Rigid bodies TCP+UDP port 4994

Road Map:



OptiTrack Tracking Tools
VRPN
Windows C Code
Socket
Python Bluetooth Code
Bluetooth
BlueSMiRF
Embedded Code

Transfer of Data:

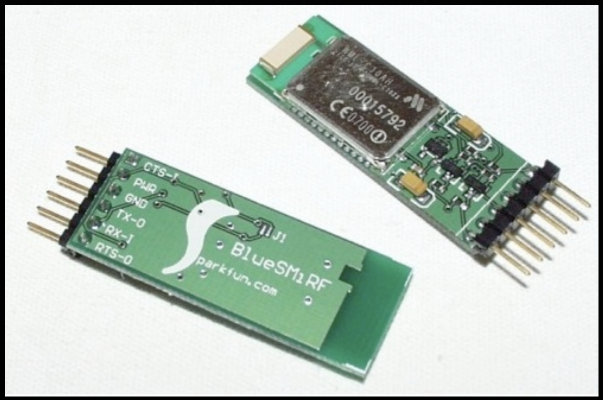
The C program reads the vrpn data from Tracking Tools over a network socket, and puts it in a data packet and sends it over a socket to python program.

We used the vrpn library: Designed to implement a network-transparent interface between application programs and a set of physical devices (tracker, etc.).

0XFF	0XFF	Length	Name	x	y	z	qx	qy	qz	qw	Check sum
Header			Data (Each is a 32 bit int)								

Python Bluetooth Code:

The python program accepts incoming connections from the blueSMiRF Bluetooth devices and sequentially broadcasts the packet with the vrpn data to all properly configured blueSMiRFs in range. The PyBluez library must be installed. This library allows us to multicast to multiple robots in the room.



Embedded Code:

The embedded program running on luminary board receives data packets and decodes them into position data.

Position Struct decoded in the Luminary's code:

Name	x	y	z	qx	qy	qz	qw	yaw	pitch	roll

We also convert the quaternion values to Euler angles.

Quaternion Values:

They are the quotient of two directed lines in a 3 dimensional space.

They are alternatives to Euler angles.