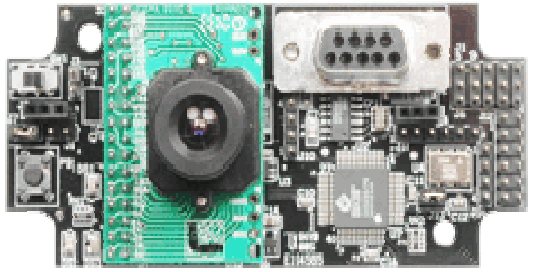
Making the camera your friend!

2007 FIRST Kickoff, Upper Darby High School

Presented by:
Frank Larkin
Lansdale Catholic Cyber Crusaders
(pafwl@aol.com)



Questions we will not answer…

- How do I write “C” code?
- How is the camera set up?
- How does the camera do its thing at the atomic, electronic and molecular level?
  - Actually, who cares?
- How is the camera connected?
- How is the camera powered?
- What the heck is a TTL serial converter?
- What is the airspeed velocity of an un-laden swallow?

Questions we will answer…

- Who is R. Kevin Watson and why should we like him?
- What does the camera basically do?
- What can I do with the camera?
- What are my camera mounting options?
- What useful information can I get from the camera?
- What can I do with this information?
- What to do ambient bright lights?
- What is YCbCr?
- When should I start?

Who is R. Kevin Watson?

- Supervisor,
  Advanced Computer Systems
  and Technologies Group
  Flight System Avionics Section
  Autonomous Systems Division
  NASA JPL – Pasadena, CA

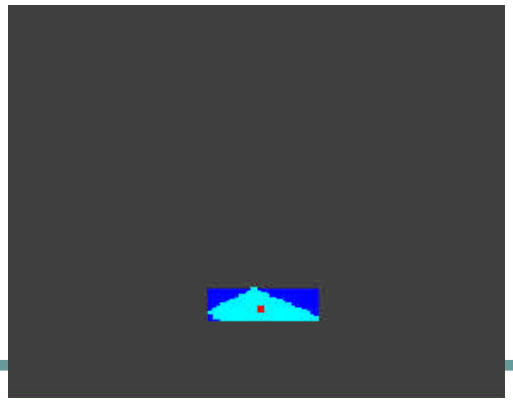
- FRC mentor (a smart guy)
- Created many code snippets for FRC robots.
  - Access his web site for all items. (www.kevin.org/frc)
  - Code examples include camera, encoders, serial i/o, gyros, and analog to digital converters. Can be a little deep.
- His stuff is out of this world, like on Mars. Yeah the planet!

What does the camera do?

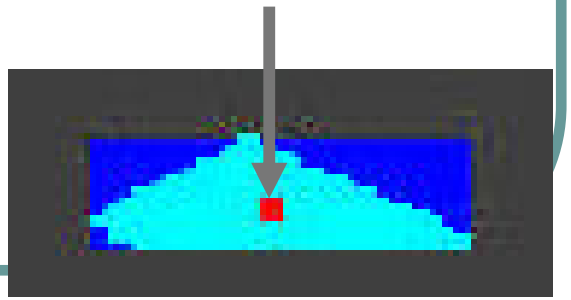
- Senses light and breaks it up into the 3 primary colors Red, Green and Blue (RGB)
- Camera can track a designated RGB value telling you where, in its field of view, the greatest color mass is.
- RGB
  - Each color has a range for 0-255
    Example: 0, 0, 0 = black 255,255,255 = white 255, 0, 0 = red

Camera Video

Camera Tracking
240, 129, 16

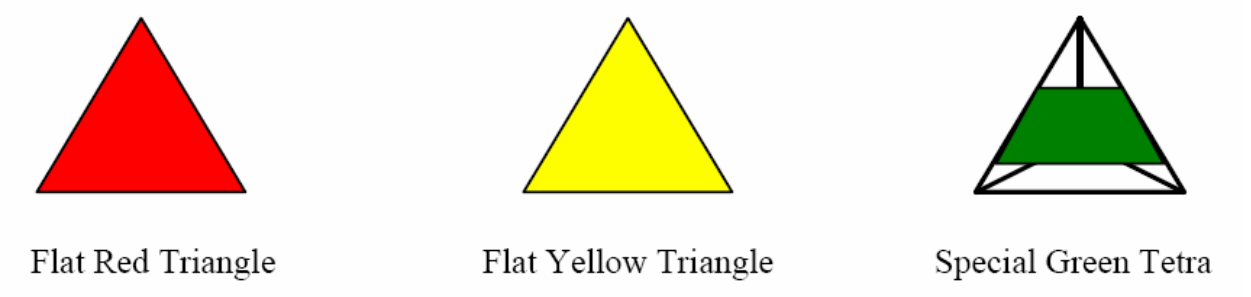


The "centroid" or middle of the mass of the color



What can I do with the camera?

- Used to see specified colors on targets.
  - adjust various devices to track specific color.
  - See if the target is in front of robot, left side or right, high or low.
- 2005 we had to track several colored triangles

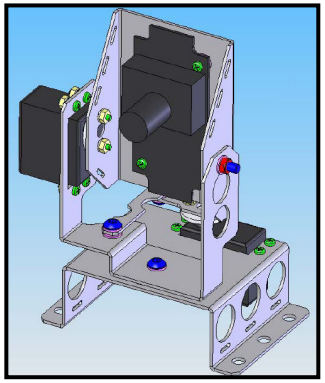


Flat Red Triangle | Flat Yellow Triangle | Special Green Tetra

  - Problems with ambient light and reflection made this difficult
- 2006 track colored green light for targeting score zone
  - Worked much better but still had problems with bright ambient light
- 2007 ?????

What are my mounting options?

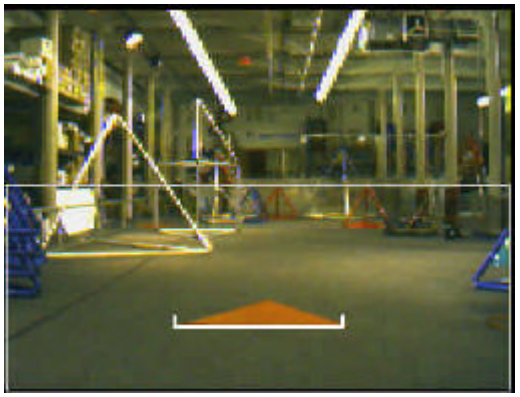
- Mount on FRC flimsy aluminum movable X & Y axis frame
  - Camera feedback is used to automatically control where camera looks
  - Plus +++
    - Can track color on its own within wide field of view
    - Searches for color when not found. Very cool!!!
  - Minus ---
    - Fragile (fra-gee-lay) One good hit, broken
    - Have been told takes time to acquire target
    - Figure out where camera is looking in relation to robot.

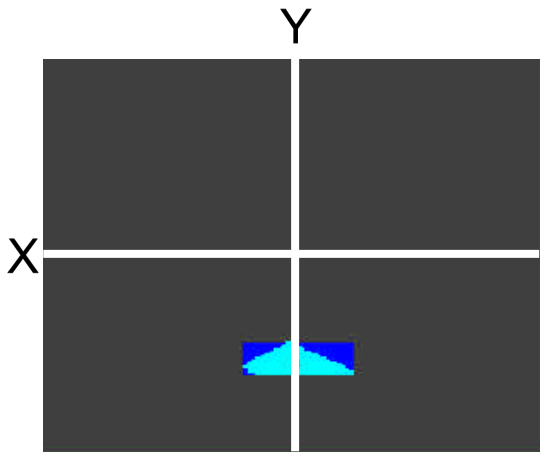


- Mount camera directly on robot (method used on 2006 LC robot)
  - Plus +++
    - Use X and Y to know where object is in camera frame
    - Y alone can tell you distance to target
    - Very little time to acquire target (if in frame)
    - Can be mounted in smaller hardened enclosure
  - Minus ---
    - Must move camera with robot component or entire robot to reacquire target

What are my mounting options?

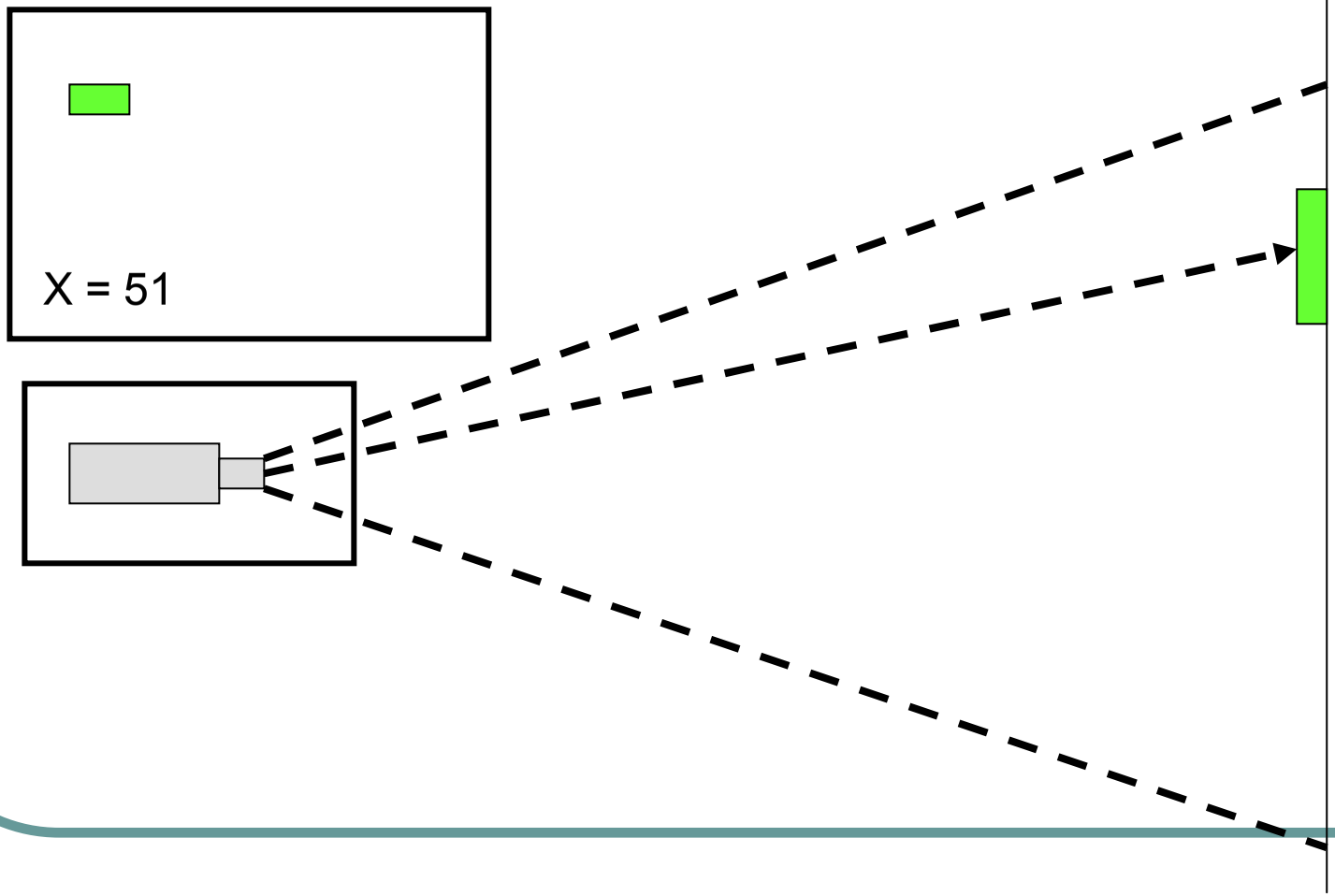
- Kevin's code was written to track using the flimsyo-alumo-auto-tracko camera frame.
- LC camera was mounted directly to robot components.
- Needed to find out if software could tell the X, Y position of the "centroid" in its field of view.
  Example: In frame below position X = 0, Y = -60





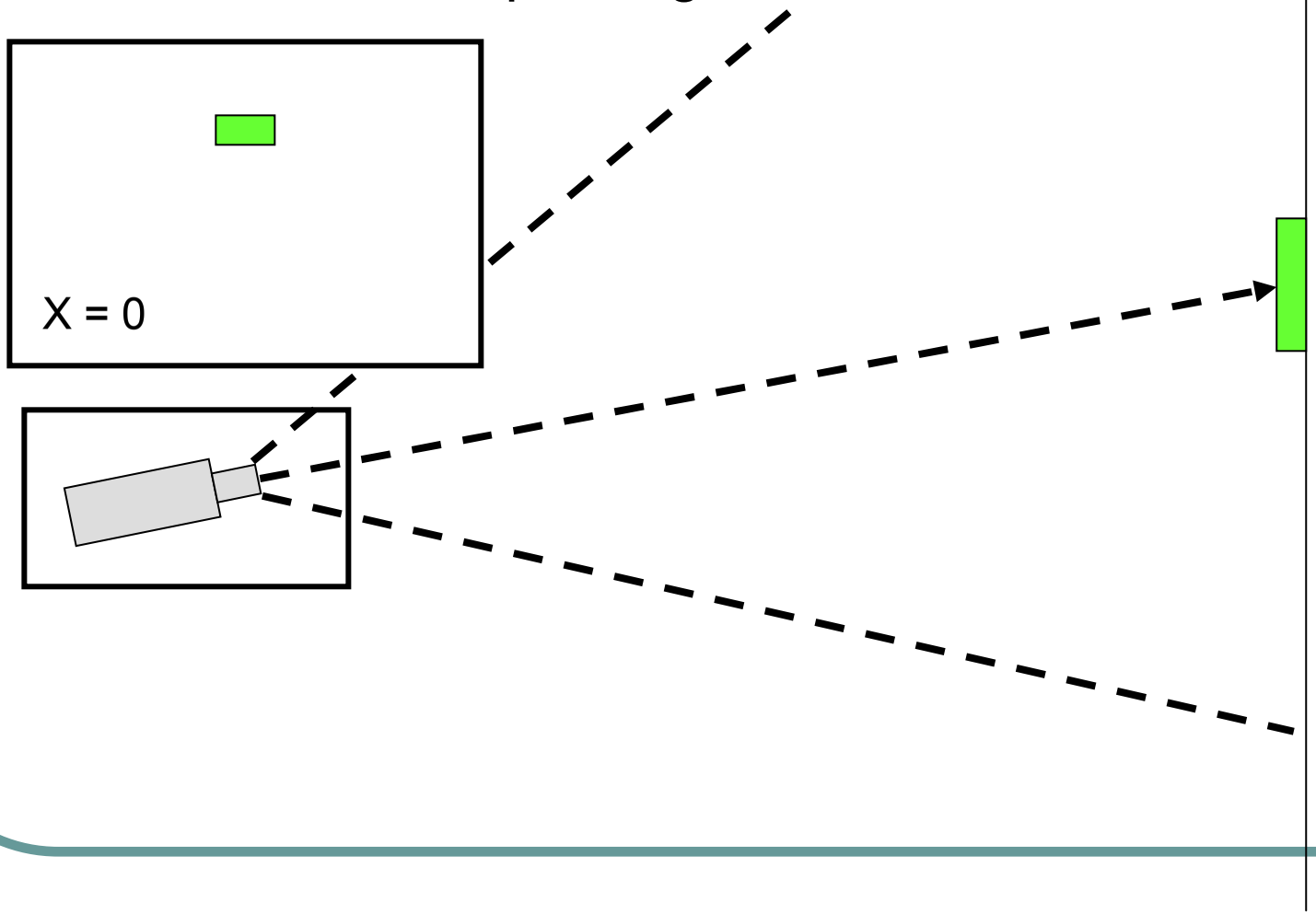
What are my mounting options?

- Camera + Shooter pan together on Shooter tower



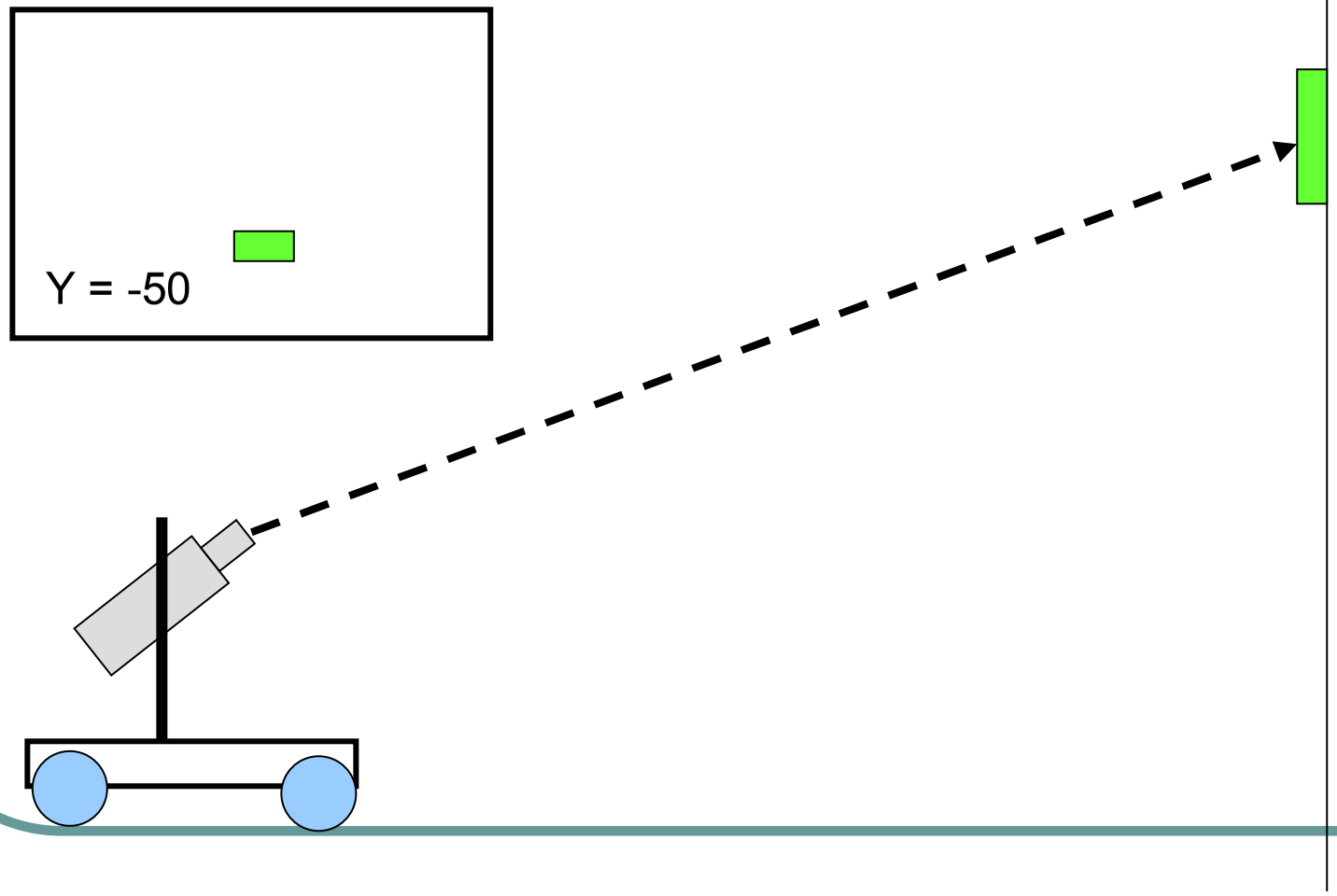
What are my mounting options?

- Camera + Shooter pan together on Shooter tower



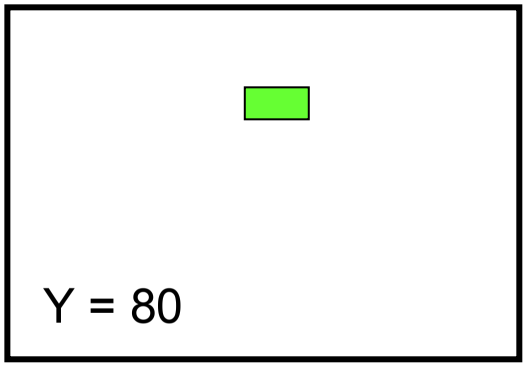
What are my mounting options?

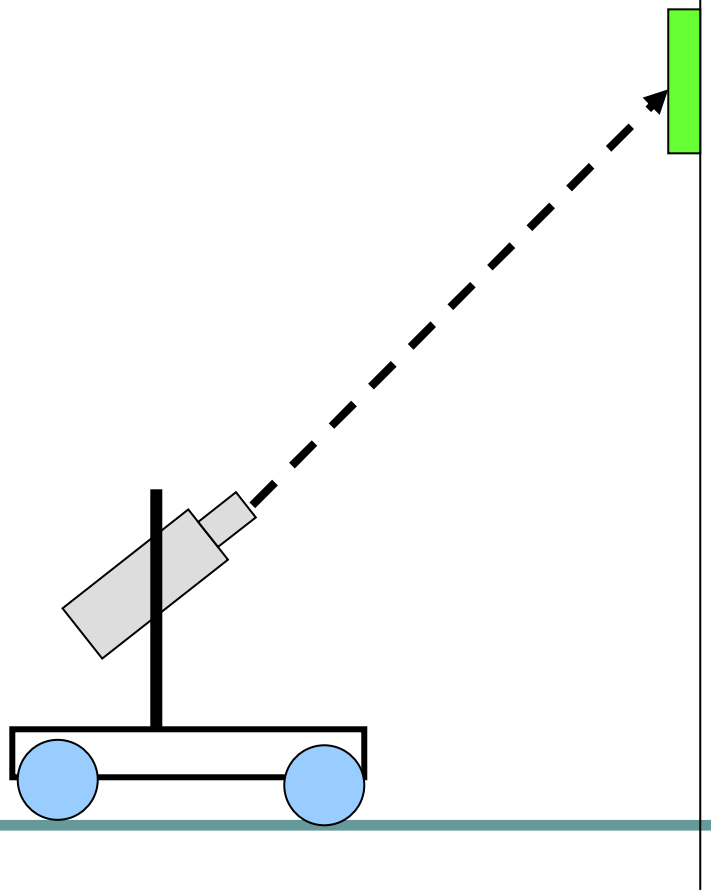
- Camera set in fixed tilt position



What are my mounting options?

- Camera set in fixed tilt position





What useful information can I get?

- Looked in Kevin's code and found where camera saves its X and Y position. Integers **T_Packet_Data.my** and **T_Packet_Data.mx** Range -127 to +127. Zero (0) = middle of frame. Created 2 global int variables to hold these. Offset by 127 to change range to 0 to 255.

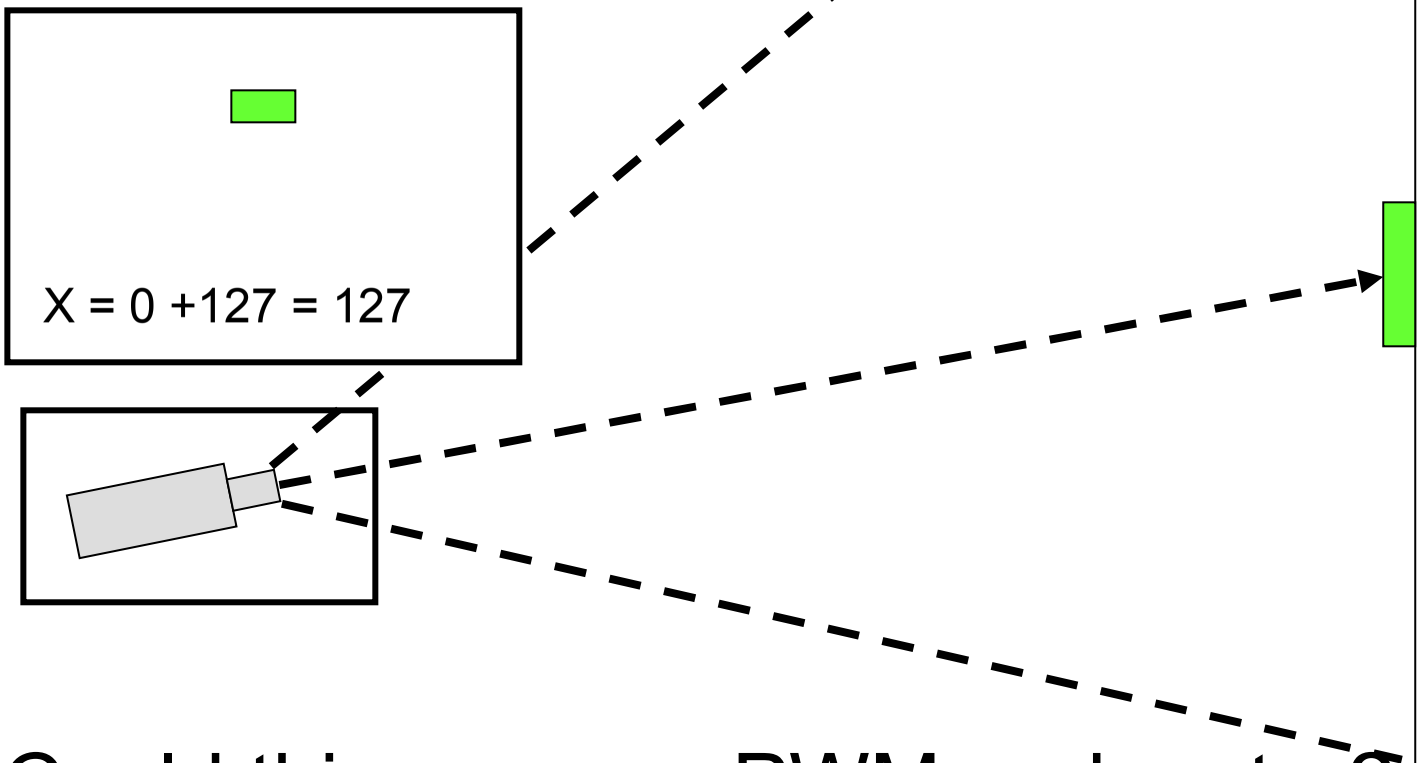
```
CameraTiltOffset =  C_OFFSET_OUT_OF_RANGE;
if( T_Packet_Data.my >= -125 && T_Packet_Data.my <= 125 )
    CameraTiltOffset =  (int) T_Packet_Data.my + 127;

CameraPanOffset  =  C_OFFSET_OUT_OF_RANGE;
if( T_Packet_Data.mx >= -125 && T_Packet_Data.mx <= 125 )
    CameraPanOffset =  (int) T_Packet_Data.my + 127;
```

- Assigned values to 2 unused PWM outputs to display in Dashboard.

```
PWM15 = (unsigned char) CameraTiltOffset;
PWM16 = (unsigned char) CameraPanOffset;
```

What useful information can I get?

- Pan center of frame now 127 (like PWM)



- Could this now run PWM and motor?

What can I do with this information?

- LC Header definitions

```
#define C_PANCENTER                     134      //Center pan value
#define C_OFFSET_OUT_OF_RANGE           255

#define C_SHOOTER_PAN_LEFT_FAST         (127-30)
#define C_SHOOTER_PAN_LEFT_SLOW         (127-8)
#define C_SHOOTER_PAN_RIGHT_FAST        (127+30)
#define C_SHOOTER_PAN_RIGHT_SLOW        (127+17)

#define C_SHOOTER_TILT_UP_FAST          (127+120)
#define C_SHOOTER_TILT_UP_SLOW          (127+15)
#define C_SHOOTER_TILT_BACK_FAST        (127-120)
#define C_SHOOTER_TILT_BACK_SLOW        (127-5)

extern int CameraTiltOffset;
extern int CameraPanOffset;
```

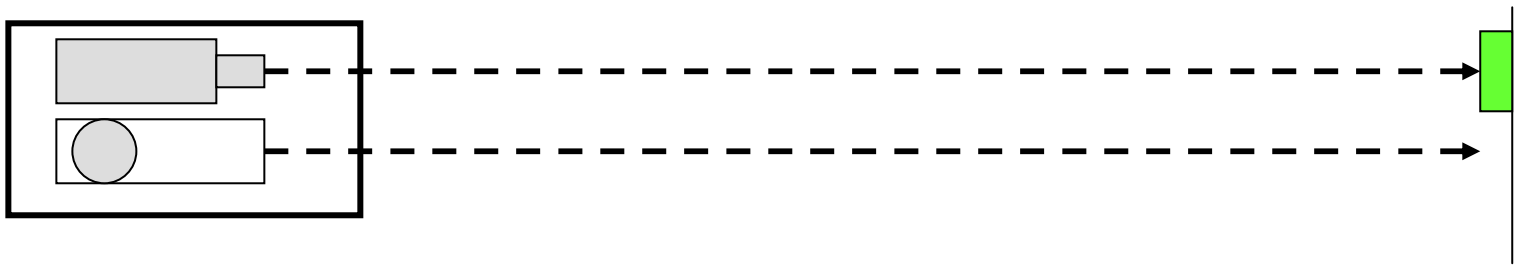
- Why use (127+offset) form?
  - Since 127 = stop, offset allows you to quickly see how much +/- motor will run.
  - What's easier (127+33) or 160?
  - Must use the parentheses but why? Look it up under #define in "C" code.

What can I do with this information?

- Why is C_PANCENTER 134 and not 127?

```
#define C_PANCENTER 134    //Center pan value
```

- Camera and Shooter not in same sight line. This offset is called “parallax”
- As you get closer parallax adjustment may be more critical. Adjustment may not even be necessary. Test, test and test



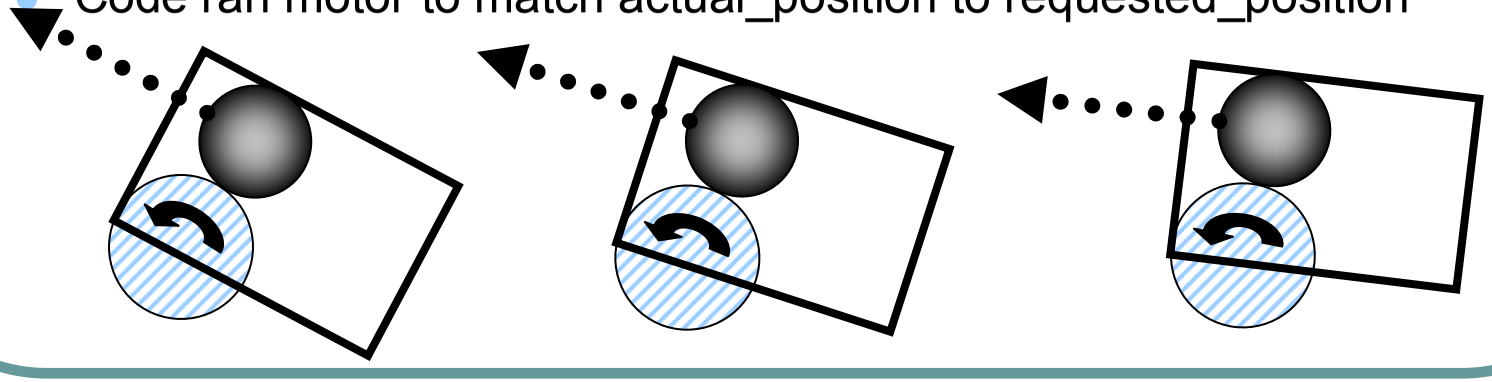
What can I do with this information?

- Shooter Tower Panning (semi-auto mode)

```
if( CameraPanOffset == C_OFFSET_OUT_OF_RANGE )
          PanPower = C_POWER_STOP;
else
{
 if( CameraPanOffset < (C_PANCENTER - 30) )
          PanPower = C_SHOOTER_PAN_RIGHT_FAST;
 else if( CameraPanOffset < C_PANCENTER )
          PanPower = C_SHOOTER_PAN_RIGHT_SLOW;

 if( CameraPanOffset > (C_PANCENTER + 30) )
          PanPower = C_SHOOTER_PAN_LEFT_FAST;
 else if( CameraPanOffset > C_PANCENTER )
          PanPower = C_SHOOTER_PAN_LEFT_SLOW;

 if( CameraPanOffset > (C_PANCENTER - 3) &&
          CameraPanOffset < (C_PANCENTER + 3) )
                  PanPower = C_POWER_STOP;
}
```

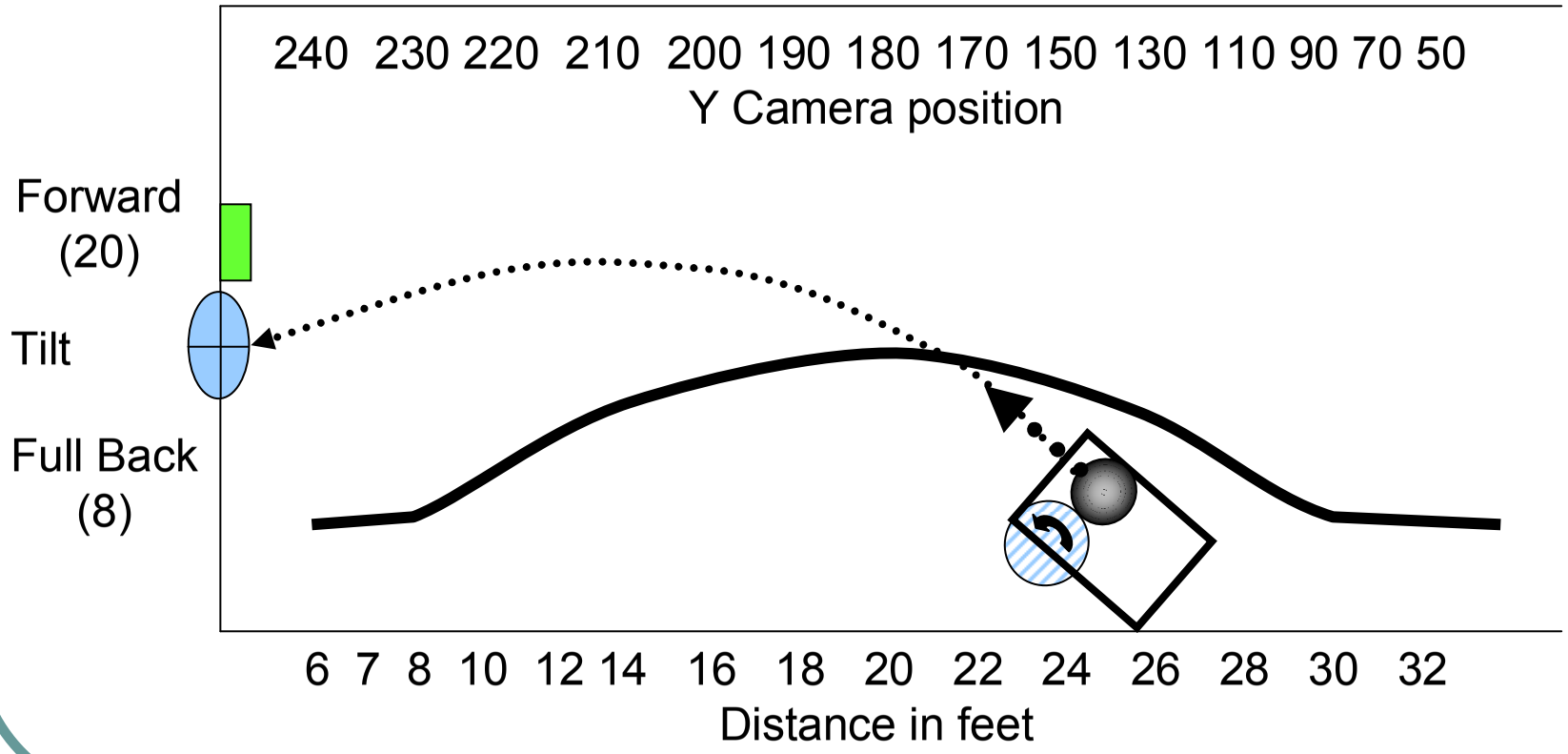
What can I do with this information?

- Shooter Tilting
  - Used lead screw and motor to adjust shooter tunnel tilt
  - Used potentiometer to sense position of shooter tunnel (actual_position)
  - Manual Mode: Used manual control panel potentiometer to input desired tilt position (requested_position)
  - Auto Mode: Use firing solution table to convert camera Y position to requested_position
  - Forced potentiometers to use range of 0-255 to display in Dashboard. Only needed values from 8 (full back, shoot high) to 22 (tilt forward, shoot low).
  - Code ran motor to match actual_position to requested_position



What can I do with this information?

- Testing showed that actual tilt followed this pattern



What can I do with this information?

- 2006 LC Robot -- Shooter Tilting

```
#Define C_SHOOTER_TILT_SENSITIVITY 3

TargetPosition = Requested_Position - Actual_Position;

if( TargetPosition < -10 )
        TiltPower = C_SHOOTER_TILT_UP_FAST;
else if( TargetPosition < (C_SHOOTER_TILT_SENSITIVITY * -1) )
        TiltPower = C_SHOOTER_TILT_UP_SLOW;

if( TargetPosition > 10 )
        TiltPower = C_SHOOTER_TILT_BACK_FAST;
else if( TargetPosition > C_SHOOTER_TILT_SENSITIVITY )
       TiltPower = C_SHOOTER_TILT_BACK_SLOW;

if( TargetPosition >= (C_SHOOTER_TILT_SENSITIVITY *-1) &&
        TargetPosition <= C_SHOOTER_TILT_SENSITIVITY)
                TiltPower = C_POWER_STOP;
```

What can I do with this information?

- Firing Solution Table (“C” switch statement)

```
RequestedPosition = GetCameraTilt();
GetCameraTilt()
{
      switch(CameraTiltPosition)
      {
            case 236: case 235: case 234: case 233: case 232:
                  return 9;
            case 231:
            case 230:
            case 229:
                  return 10;
            default:
                  return 8;       // full tilt back
      }
}
```

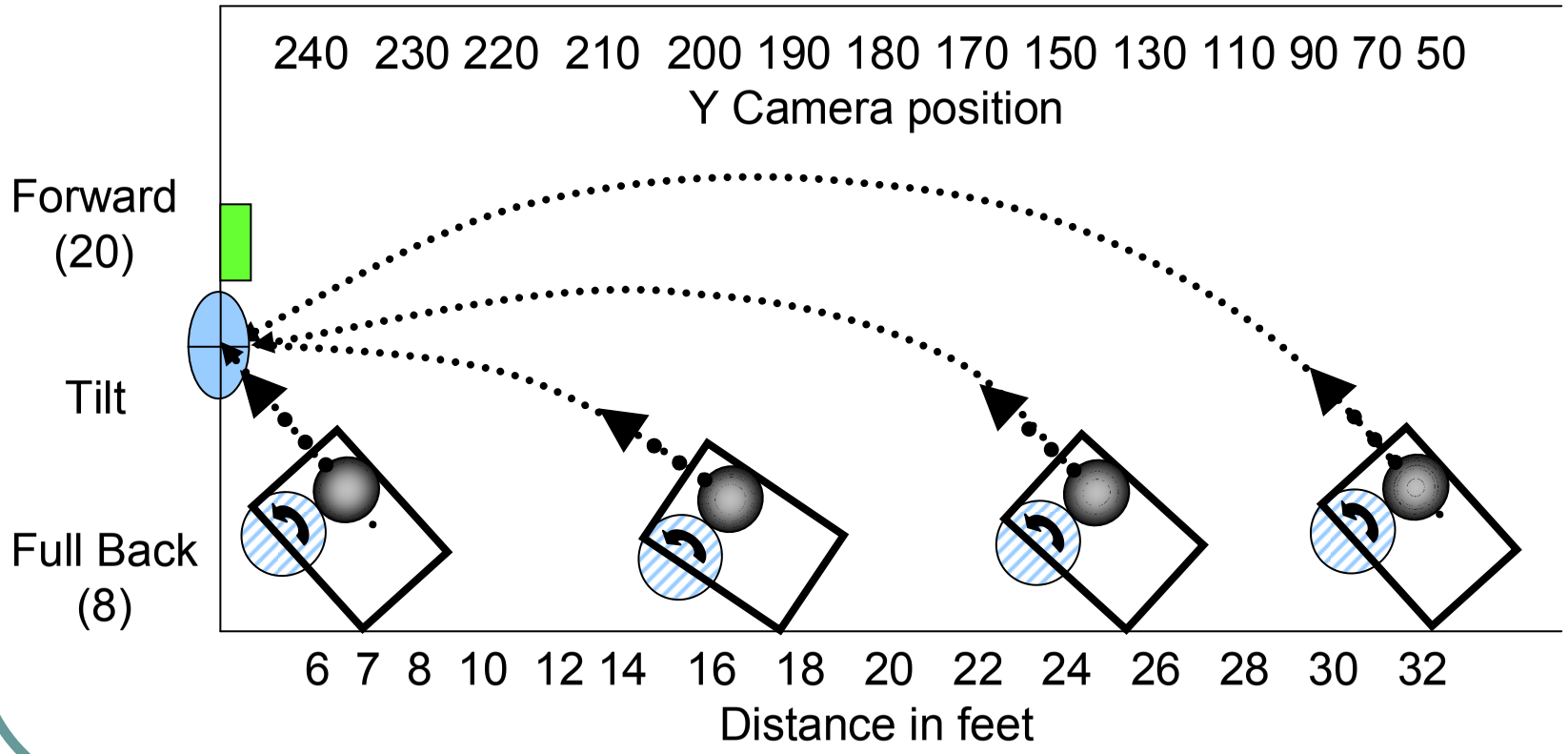
What can I do with this information?

- Alternate firing solution array

```
// these are the settings of the shooter tilt based upon
// input from the camera (range 250-50). We divide the
// camera tilt offset by 10 giving us the index below.
int ShootTilt[ ] = {  8,  //0 - required because arrays are zero based, element 1 is index 0
                      8,   8,   8,   8,   8,    //  1 to  5, represents camera tilt 10-50
                      8,   8,  20, 22,  20,    //  6 to 10, represents camera tilt 60-100
                     20, 20, 20, 19,  14,    // 11 to 15, represents camera tilt 110-150
                     14, 14, 12, 12,   9,     // 16 to 20, represents camera tilt 160-200
                      9,  9,  8,  8,   8,      // 21 to 25, represents camera tilt 210-250
                      8,  8,  8,  8,   8 };    // 26 to 30, represents camera tilt 260-300

printf( “Shooter tilt = %d\r”, ShooterTilt[10] );   // what does this print?
```

What can I do with this information?

- Adjusted trajectory to get ball in goal.



What to do about ambient bright lights?

- Too high an exposure setting can wash out target light and ambient light making them look like the same RGB color. Example: 240, 240, 240
- Running camera at lower exposure cuts down total light coming in to camera. All you want to see is the target light's color even if it is at reduced levels.
- Test with bright white and other colored lights near target. See how close you can come with interfering colored lights near to the target light without effecting targeting. Try bright white light right behind target light.
- Two seconds after you get into the competition venue check for brightly colored lights, other robot's lights, other robot's colors or team banners that might interfere. Ask for these to be covered or removed. If you can prove they are a problem make a big Gracious Professional stink about it. Still no response, Ask the question, "Why did FIRST bother having the camera when it cannot be used?"
- Test, test, test and test some more.

What is YCbCr?

- **YCbCr** is a family of color spaces used in video systems. Y is the luma component and Cb and Cr are the blue and red chroma components.
- The camera can get you results in RBG or in YCbCr. Make sure that the test software outside the robot (like in LabView) be set the same as the Robot's software. YCbCr was the default of Kevin's code in the past.
- This also applies to other settings, auto exposure, bla, bla, bla
- When you go to a competition they may supply settings for everyone. Ask how these were determined.
- Not a big deal just be aware of it.

Settings in camera.h…

```
#define R_MIN_DEFAULT       85      // Rmin for call to Track_Color()
#define G_MIN_DEFAULT       15      // Gmin for call to Track_Color()
#define B_MIN_DEFAULT       100     // Bmin for call to Track_Color()

#define R_MAX_DEFAULT       115     // Rmax for call to Track_Color()
#define G_MAX_DEFAULT       16      // Gmax for call to Track_Color()
#define B_MAX_DEFAULT       145     // Bmax for call to Track_Color()
#define NF_DEFAULT          0       // value for call to Noise_Filter()
#define AGC_DEFAULT         0       // Automatic Gain Control Register [0/0x00]
#define BLU_DEFAULT         128     // Blue Gain Control Register [128/0x80]
#define RED_DEFAULT         128     // Red Gain Control Register [128/0x80]
#define SAT_DEFAULT         128     // Saturation Control Register [128/0x80]
#define BRT_DEFAULT         1       // Brightness Control Register 128/0x80]
#define AEC_DEFAULT         1       // Automatic Exposure Control Register [127/0x7F]
#define COMA_DEFAULT        32      // Common Control A Register [36/0x24]
#define COMB_DEFAULT        32      // Common Control B Register [1/0x01]
#define COMI_DEFAULT        128     // Common Control I Register [0/0x00]
#define EHSH_DEFAULT        128     // Frame Rate Adjust Register 1 [0/0x00]
#define EHSL_DEFAULT        32      // Frame Rate Adjust Register 2 [0/0x00]
#define COMJ_DEFAULT        132     // Common Control J Register [129/0x81]
```

When should I start?

- Today or immediately if not sooner which ever comes first.
- Assign a dedicated group to work on camera and all sensors.
- Create a test bed to play with the camera and all sensors.
- Suggestion: Don’t tell everyone how great it will be, just do it.
- Anyone can talk the talk, you walk the walk.