Control Testbed

Student: Ezra Idy
Professor: Vikram Kapila

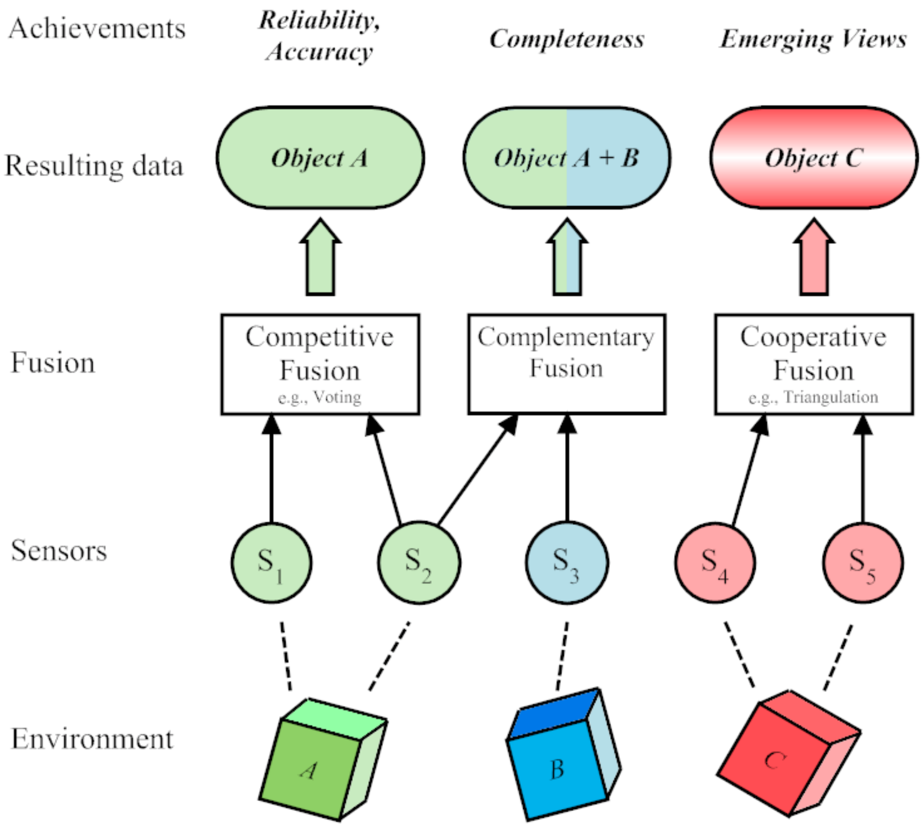
Background

What is Sensor Fusion?

The combination of data from several sensors for the purpose of improving application or system performance.

3 distinct types:

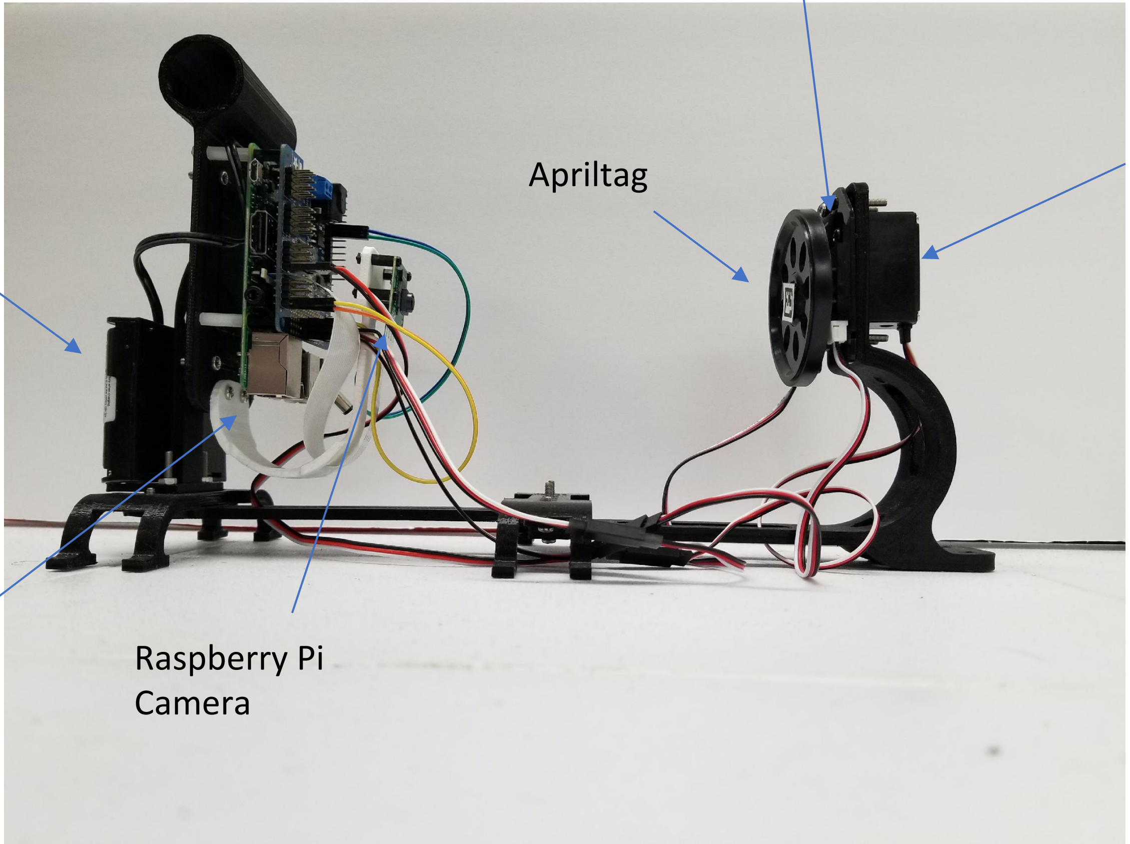
- Competitive Fusion – independent measurements of the same property
- Complementary Fusion – more complete view of object
- Cooperative Fusion – derived information to obtain completely new information



Objective

- Create an easy, affordable, and accessible Sensor Fusion system
- Analyze the data obtained from:
  - Individual sensors
  - Multiple sensors

Testbed: First Design



Battery Pack

Raspberry
Pi
&
Shield

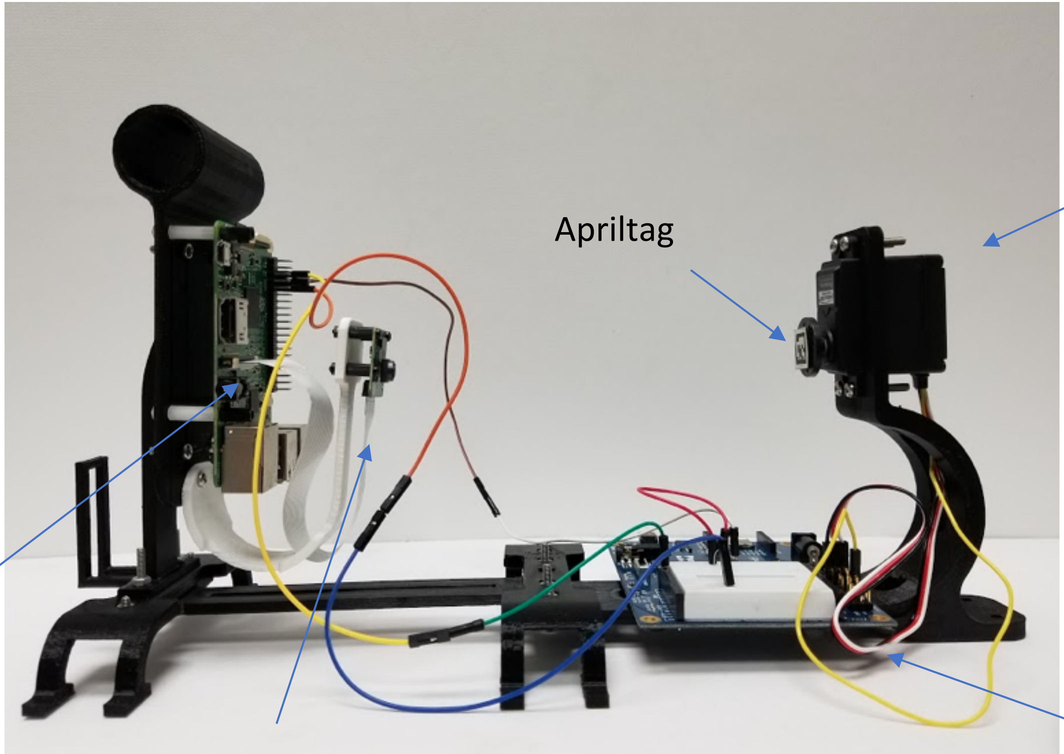
Raspberry Pi
Camera

Apriltag

8-bit encoder

Parallax
Continuous Servo
Motor

Testbed: Second Design



Parallax Feedback 360
High Speed Servo
Motor

Apriltag

Raspberry Pi

Raspberry Pi
Camera

Parallax
Propeller

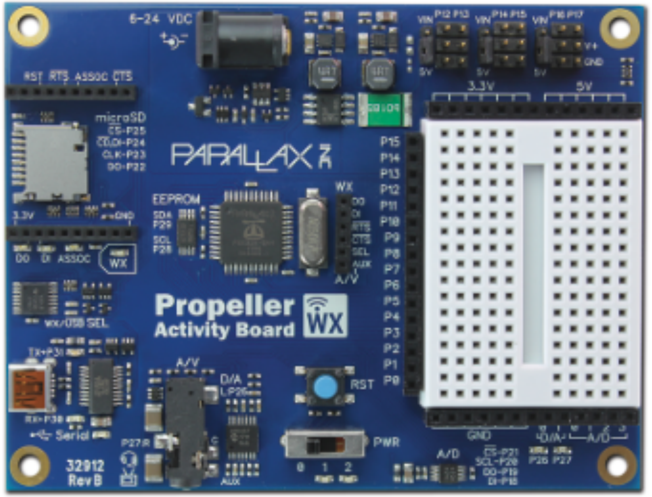
Cost

Parts	Amount	First Design	Second Design
Raspberry Pi 3	1	$40	$40
Parallax Propeller Board	1	$0	$71
Raspberry Pi Camera	1	$26	$26
Servo Motor	1	$15	$28
Servo/PWM Pi HAT	1	$17	$0
Miscellaneous	1	$70	$70
Total	------	$168	$235

- Universal Power Module Model No. UPM 1503: $374.98
- Quanser Consulting Plant SRV-02 Tachometer + Amenities : $9,171.48

Parallax Propeller

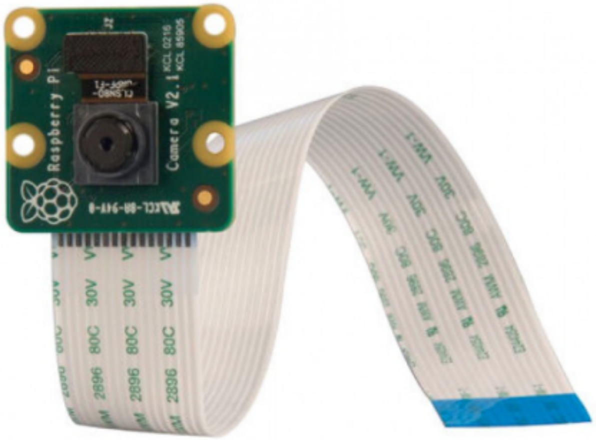
- 8-core Propeller microcontroller and 64 KB EEPROM
  - 5 cogs used for this project
- 3 position power switch
- 16 programmable GPIO pins
  - P0 - P15




```
//--------- Cogs---------------
void test(void *par);          // cog for control
void Feedback360();            // cog for feedback
void Serial1();                // cog for communication with ios platform
void Serial2();                // cog for recieving continuous camera angle
//---------------------------
```

Camera

- The Camera is used to detect the AprilTags orientation
  - Converted Quaternion into Roll, Pitch, Yaw
- The rate of the Camera is 60 Hz
- High resolution images
- Slow but accurate
- Lighting issues when detecting



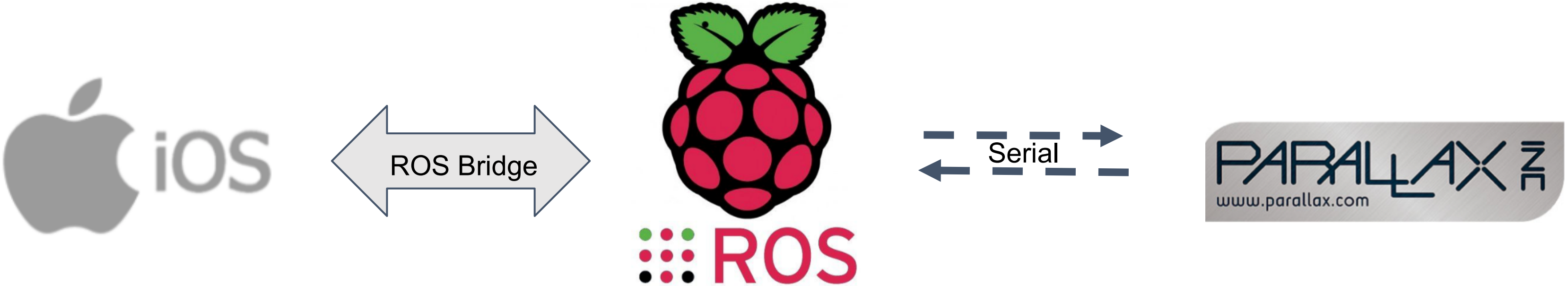
Motor

- The motor is a continuous servo motor with a feedback pin
  - Uses internal Hall effect sensors
  - No need to “center” the motor
- Feedback signal: PWM, 3.3V, 910 Hz, 2.7%–97.1% duty cycle
- Low load rotation from -120 to 120 RPM
- Peak stall torque @ 6 V: 2.5 kg-cm (34.7 oz-in)

Application

- Educational purpose – Can be used to teach sensor fusion on a basic level
- Interactive learning – Learn Kalman Filtering and PID control
- Embedded vs remote – Learn about the different sensor types and how they affect a system

Communication



Serial Communication 1

- Communication is done through the USB serial port
- Data is given by user
  - Processed and converted into a byte array
- Once on the Propeller end, the data is converted back to the appropriate values
- For angles two bytes are sent
  - Bit shifting is needed

```
com = fdserial_open(31, 30, 0, 115200);

void Serial1()
{
  char a[SIZE];
  while(1)
  {
    int b = fdserial_rxReady(com);
    if (b != 0){
      a[0] = fdserial_rxChar(com);
      //dprint(com, "a[0] = %d\n", a[0]);
      for(int i = 1; i <SIZE; i++)
      {
        a[i] = fdserial_rxChar(com);
        //dprint(com, "a[%d] = %d\n",i, a[i]);
      }
    }
    y = a[0]; //a[0]
    //dprint(com, "bool value = %d\n", y);
    targetAngle = (unsigned)(a[1]<<8) + a[2];
    dprint(com, "targetAngle = %d\n", targetAngle);
    speed = (signed char)a[3];
    //dprint(com, "speed value = %d\n", speed);
    Kp = a[4]/10.0;
    //dprint(com, "kp value = %f\n", Kp);
    Ki = a[5]/10.0;
    //dprint(com, "ki value = %f\n", Ki);
    Kd = a[6]/10.0;
    //dprint(com, "kd value = %f\n", Kd);
    motorK = a[7];
    cameraK = a[8];
    fuseSwitch = a[9];
    motorF = a[10]/10.0;
    cameraF = a[11]/10.0;
```

Serial Communication 2

- Communication is done through GPIO
- Seperate channel to send the camera data
- Bit shifting is used

```
ros = fdserial_open(7, 8, 0, 115200);
```

```
void Serial2()
{
  char ang[LENGTH];
  while(1)
  {
    int b = fdserial_rxReady(ros);
    if (b != 0)
    {

      for(int i = 0; i < LENGTH; i++)
      {
        dprint(ros, "Reading from ros\n");
        ang[i] = fdserial_rxChar(ros);
      }
      fdserial_rxFlush(ros);
      aprilTag = (unsigned)((ang[0]<<8) + ang[1]);
      dprint(ros, "does it work? %d\n", aprilTag);

    }
    //dprint(ros, "does it work? %d\n", aprilTag);
    pause(100);
  }
}
```

PID Control

- Allow for move to target control
- Output is experimentally tested and maxed at ± 120
- Due to restrictions in the serial communication, all gains are limited from 0 to 25
- Incorporated in the app

```
void PID(int x, bool y)
{
  if(y == 1)
  {
    //  dprint(com, "x = %d\n", x);
    errorAngle = x - angle;          // Calculate error
   // dprint(com, "error: %d\n", errorAngle);
    integral = integral + errorAngle;
    derivative = errorAngle - last_err;
    last_err = errorAngle;
    output = (errorAngle * Kp) + (integral * Ki) + (derivative *Kd);
   // dprint(com, "output: %d\n", output);
    if(output > 120) output = 120;           // Clamp output, Max Value
    if(output < -120) output = -120;         // Clamp output, Min Value
    // An offset can be added if needed, to account for specific motor capabilities
    // However this code is not using an offset
    servo_speed(pinControl,  output);

  }
}
```

Kalman Filtering

- Implement Kalman Filtering that will add weighted values to the different data obtained
- The noise Q and R were arbitrarily selected
  - Fine tuning may be needed
- Incorporated in the app

```
float Kalman(int data)
{
  P_temp = P_last + Q;
  Kgain = P_temp/(P_temp + R);
  //current_est = prev_est + Kgain*(data - prev_est);
  current_est = Kgain*data + prev_est*(1-Kgain);
  P_current = (1 - Kgain)*P_temp;
  //dprint(com, "Kgaig = %d\r", P_0);

  P_last = P_current;
  prev_est = current_est;

  return current_est;
}
```

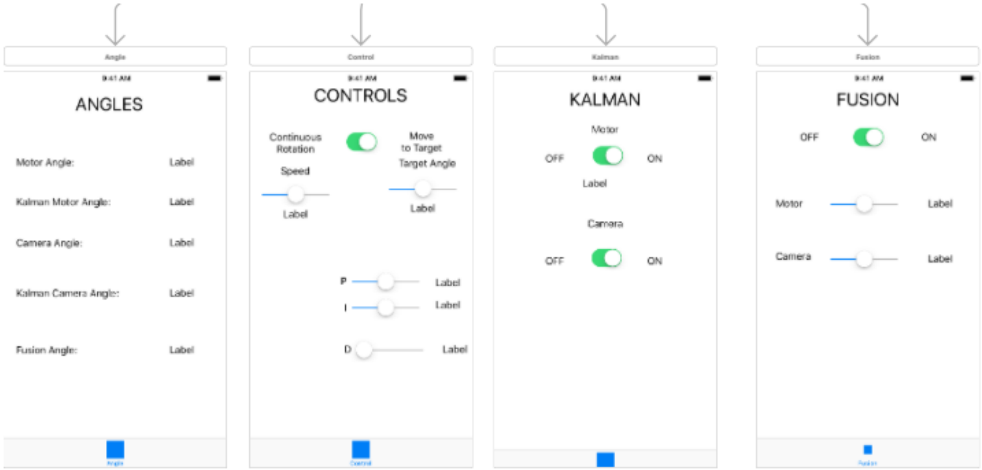
Fusion

- Applying competitive fusion
- Interactive voting depending on reliability of the sensor
- Incorporated in the app

```
float Fusion(int m, int c)
{
  float fused = ((m*motorF)+(c*cameraF))/(motorF+cameraF);
  return fused;
}
```

Graphic User Interface

- Redesigned GUI
  - Tabbed application
- Connect to ROS through ROSBridge
- Allow for the selection of options:
  - Fusion
  - PID
  - Kalman
- Easy testing
- Can observe the data from the app



Future Work

- Take a video

Entrepreneurship

- Project was chosen for the Stern Entrepreneurship collaboration
- The Stern students are currently validating customers
- Testing the educational marketplace
  - In contact with CIJE (Center for Initiatives in Jewish Education)
  - In contact with ITEST