Rewrite of the Vision Tracking System, To Benefit the Accuracy and Speed of the nVidia Tegra©

Nicholas Burrell
Software Team

Febuary 28, 2016

1 Introduction

1.1 Before We Begin

Thank you for taking the time to read the documentation for our systems new rewrite. The old system had a ***LOT*** of issues, which we can see below. They range from simple things like inconsistent variable naming schemes and tabbing, to more extreme things like memory management and even neglecting an entire subsystem of the Tegra©, the CUDA© Cores.
Some examples of poor writting include...

```
 1 template <class T>
 2 T getMin (vector <T> input) {
 3    size_t i;
 4   T minimum = 0;
 5   T minimum_out = 0;
 6    if (input.size() == 0)
 7      return 360;
 8    if (input.size() == 1)
 9      return input[0];
10    else{
11      minimum = 360;
12      for (i = 0; i < input.size(); i++) {
13        if (minimum > abs(input[i])) {
14          minimum = abs(input[i]);
15          minimum_out = input[i];
16        }
17      }
18    }
19    return minimum_out;
20 }
```

Listing 1: Poor Coding Part 1

The first issue with this example, is it is written for reusability, while it doesn't need to be. That is *not* and issue, but for the sake of simplicity, it can and *will* be rewritten. The second being the hard coded value. Line 11, to be precise. If we were to write this code to be reusable, we should just have the value be some arbitrarily large number, say, 600 billion. Speaking of bad practices...

```

float angle;
  int numOfBadFrames = 0;
    bool die = false;
  string filename("snapshot");
  string suffix(".png");
  int i_snap(0), iter(0);
  auto index = 0;
    vector < vector <Point> > contours;
    vector<Rect > boundRect;
  vector <KeyPoint> keypoints;
  vector <Point> centers;
  //vector <float > abs_angle;
  Mat frame(Size(480, 360), CV_8UC3, Scalar(0));
  bool bIsConnected = false;
  int sockfd, new_fd;  // listen on sock_fd, new connection on new_fd
  struct addrinfo hints, *servinfo, *p;
  struct sockaddr_storage cli_addr; // connector's address information
  socklen_t sin_size;
  struct sigaction sa;
  int yes=1;
  char s[INET6_ADDRSTRLEN];
  int rv;
  char buf[256];
  int numchars = 0;
  vector<float > angles;
  Mat LastFrame(Size(480, 360), CV_8UC3, Scalar (0));
```

Listing 2: Poor Coding Part 2

I mean what is up with the naming scheme. It appears as though the code was copied off of the internet. Granted, a small portion of it was. Mostly, the sockets. Along with this, the tabbing is atrocious. This code is *begging* to be rewritten.

1.2 General Improvements

Before we delve into what will be rewritten, lets compare and contrast some of the issues with solutions.
The first issue is the writing style. Relatively easy fix. The next issue, is speed. The fact that the camera in the rear of the robot can go at *whatever* framerate I need at the time, means my code should perform that fast as well. Parallel processing would improve the speed *drastically*, as would the GPU. Some of these will require some things like named pipes, or more advanced things such as CUDA mathematics.