branch: master AI-Shack--Tracking-with-OpenCV / TrackColour.cpp

liquidmetal on Jun 17, 2011 Uses OpenCV 2.2 now

1 contributor

file 116 lines (89 sloc) 3.047 kb Edit Raw Blame History Delete

```
// TrackColour.cpp : Defines the entry point for the console application.
//

#include "stdafx.h"

#include <opencv2\opencv.hpp>

IplImage* GetThresholdedImage(IplImage* img)
{
	// Convert the image into an HSV image
	IplImage* imgHSV = cvCreateImage(cvGetSize(img), 8, 3);
	cvCvtColor(img, imgHSV, CV_BGR2HSV);

	IplImage* imgThreshed = cvCreateImage(cvGetSize(img), 8, 1);

	// Values 20,100,100 to 30,255,255 working perfect for yellow at around 6pm
	cvInRangeS(imgHSV, cvScalar(112, 100, 100), cvScalar(124, 255, 255), imgThreshed);

	cvReleaseImage(&imgHSV);

	return imgThreshed;
}

int main()
{
	// Initialize capturing live feed from the camera
	CvCapture* capture = 0;
	capture = cvCaptureFromCAM(0);

	// Couldn't get a device? Throw an error and quit
	if(!capture)
    {
	printf("Could not initialize capturing...\n");
	return -1;
    }


	// The two windows we'll be using
    cvNamedWindow("video");
	cvNamedWindow("thresh");

	// This image holds the "scribble" data...
	// the tracked positions of the ball
	IplImage* imgScribble = NULL;

	// An infinite loop
	while(true)
    {
		// Will hold a frame captured from the camera
		IplImage* frame = 0;
		frame = cvQueryFrame(capture);

		// If we couldn't grab a frame... quit
	if(!frame)
	    break;

		// If this is the first frame, we need to initialize it
		if(imgScribble == NULL)
		{
			imgScribble = cvCreateImage(cvGetSize(frame), 8, 3);
		}

		// Holds the yellow thresholded image (yellow = white, rest = black)
		IplImage* imgYellowThresh = GetThresholdedImage(frame);

```

```
65                  // Calculate the moments to estimate the position of the ball
66                  CvMoments *moments = (CvMoments*)malloc(sizeof(CvMoments));
67                  cvMoments(imgYellowThresh, moments, 1);
68
69                  // The actual moment values
70                  double moment10 = cvGetSpatialMoment(moments, 1, 0);
71                  double moment01 = cvGetSpatialMoment(moments, 0, 1);
72                  double area = cvGetCentralMoment(moments, 0, 0);
73
74                  // Holding the last and current ball positions
75                  static int posX = 0;
76                  static int posY = 0;
77
78                  int lastX = posX;
79                  int lastY = posY;
80
81                  posX = moment10/area;
82                  posY = moment01/area;
83
84                  // Print it out for debugging purposes
85                  printf("position (%d,%d)\n", posX, posY);
86
87                  // We want to draw a line only if its a valid position
88                  if(lastX>0 && lastY>0 && posX>0 && posY>0)
89                  {
90                          // Draw a yellow line from the previous point to the current point
91                          cvLine(imgScribble, cvPoint(posX, posY), cvPoint(lastX, lastY), cvScalar(0,255,255), 5);
92                  }
93
94                  // Add the scribbling image and the frame... and we get a combination of the two
95                  cvAdd(frame, imgScribble, frame);
96                  cvShowImage("thresh", imgYellowThresh);
97                  cvShowImage("video", frame);
98
99                  // Wait for a keypress
100                 int c = cvWaitKey(10);
101                 if(c!=-1)
102                 {
103                         // If pressed, break out of the loop
104             break;
105                 }
106
107                 // Release the thresholded image... we need no memory leaks.. please
108                 cvReleaseImage(&imgYellowThresh);
109
110                 delete moments;
111     }
112
113         // We're done using the camera. Other applications can now use it
114         cvReleaseCapture(&capture);
115     return 0;
116 }
```