**Computer Vision for Assembly Line**

**(CVAL)**

**Source Code**

**Source Code**



#include "opencv2/core/core.hpp"

#include "opencv2/highgui/highgui.hpp"

#include "opencv2/features2d/features2d.hpp"

#include "opencv2/nonfree/features2d.hpp"

#include "opencv2/legacy/legacy.hpp"

#include "opencv2/video/tracking.hpp"

#include <stdio.h>

#include <string>

#include <conio.h>

#include <iostream>

#include <cstring>

/\*

\* Defining Constants for display and trackbars

\*/

#define MAX\_CONTOUR\_LEVELS 10

#define IMG\_WIDTH 320

#define IMG\_HEIGHT 240

#define DEFAULT\_TRACKBAR\_VAL 128

int main(int argc, const char \* argv[]) {

/\*

\* Creating variables to store image data and related values

\*/

char quit = 0;

char pass[10],user[10];

char grab\_frame = 1;

double area1,area2;

int errcount = 0;

int thresh1 = DEFAULT\_TRACKBAR\_VAL, thresh2 = DEFAULT\_TRACKBAR\_VAL;

cvNamedWindow("Template", 0);

IplImage \*small\_image = cvCreateImage(cvSize(IMG\_WIDTH, IMG\_HEIGHT), IPL\_DEPTH\_8U, 3);

IplImage \*small\_grey\_image = cvCreateImage(cvGetSize(small\_image), IPL\_DEPTH\_8U, 1);

IplImage \*edge\_image = cvCreateImage(cvGetSize(small\_image), IPL\_DEPTH\_8U, 1);

IplImage \*frame\_template= cvCreateImage(cvSize(IMG\_WIDTH, IMG\_HEIGHT), IPL\_DEPTH\_8U, 3);

CvMemStorage \*storage = cvCreateMemStorage(0);

CvSeq \*contours = 0,\*template\_contours=0;

/\*

\* Loop to validate password;

\*//\*

while (true)

{

std::cout << "Enter Username";

std::fgets(user, 6, stdin);

std::fflush(stdin);

std::cout << user << "\n";

std::cout << "Enter Password";

std::fgets(pass, 5, stdin);

std::fflush(stdin);

std::cout << pass << "\n";

if (strcmp(pass, "pass") || strcmp(user, "admin"))

{

std::cout << "Incorrect Username/Password\n";

}

else

break;

}

\*/

/\*

\* Loading and processing Template

\*/

frame\_template = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowTEMP.jpg");

cvResize(frame\_template, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &template\_contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, template\_contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Template", small\_image);

//CvCapture \*camera = cvCreateCameraCapture(0);

//if (!camera) {

// printf("Could not find a camera to capture from...\n");

// return -1;

//}

cvNamedWindow("Process\_OutputA", 0);

cvNamedWindow("Process\_OutputB", 0);

cvNamedWindow("Process\_OutputC", 0);

cvNamedWindow("Process\_OutputD", 0);

cvNamedWindow("Process\_OutputE", 0);

cvNamedWindow("Process\_OutputF", 0);

cvNamedWindow("Process\_OutputG", 0);

cvNamedWindow("Process\_OutputH", 0);

cvNamedWindow("Process\_OutputI", 0);

cvNamedWindow("Process\_OutputJ", 0);

cvDestroyAllWindows();

cvCreateTrackbar("Thresh1", "Process\_OutputA", &thresh1, 256, 0);

cvCreateTrackbar("Thresh2", "Process\_OutputA", &thresh2, 256, 0);

cvSetTrackbarPos("Thresh1", "Process\_OutputA", DEFAULT\_TRACKBAR\_VAL);

cvSetTrackbarPos("Thresh2", "Process\_OutputA", DEFAULT\_TRACKBAR\_VAL);

/\*

\* Loading and processing Samples

\*/

while (!quit) {

IplImage \*frame;

int c = cvWaitKey(30);

switch (c) {

case 32:

grab\_frame = !grab\_frame;

break;

case 27:

quit = 1;

break;

};

if (!grab\_frame)continue;

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowA.jpg");

if (!frame)continue;

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputA", small\_image);

}

/\*

\* Checking for defects

\*/

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

std::fflush(stdin);

scanf("");

cvDestroyAllWindows();

IplImage \*frame;

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowB.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputB", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*

frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowC.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputC", small\_image);

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowD.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputD", small\_image);

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowE.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputE", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowF.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputF", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowG.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputG", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowH.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputH", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowI.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputI", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*frame = cvQueryFrame(camera);\*/

frame = cvLoadImage("C:\\Users\\user\\Desktop\\SE\\arrowJ.jpg");

cvResize(frame, small\_image, CV\_INTER\_LINEAR);

cvCvtColor(small\_image, small\_grey\_image, CV\_RGB2GRAY);

cvCanny(small\_grey\_image, edge\_image, (double)thresh1, (double)thresh2, 3);

cvDilate(edge\_image, small\_grey\_image, 0, 1);

cvFindContours(small\_grey\_image, storage, &contours, sizeof(CvContour), CV\_RETR\_TREE, CV\_CHAIN\_APPROX\_NONE, cvPoint(0, 0));

cvDrawContours(small\_image, contours, CV\_RGB(255, 0, 0), CV\_RGB(0, 255, 0), MAX\_CONTOUR\_LEVELS, 1, CV\_AA, cvPoint(0, 0));

cvShowImage("Process\_OutputJ", small\_image);

area1 = fabs(cvContourArea((CvContour \*)contours, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Dimention\n");

errcount++;

}

if (contours->v\_next->v\_next->h\_next)

{

printf("Improper formation of block: Defect\n");

errcount++;

}

area1 = fabs(cvContourArea((CvContour \*)contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

area2 = fabs(cvContourArea((CvContour \*)template\_contours->v\_next->v\_next, CV\_WHOLE\_SEQ));

if (area1 != area2)

{

printf("Improper formation of block: Slot Dimention\n");

errcount++;

}

cvWaitKey(0);

cvDestroyAllWindows();

/\*Defect percentage calculator\*/

if(errcount/10>0.30)

printf("WARNING: HIGH AMOUNT OF ERRORS DETECTED IN BATCH\n");

std::fflush(stdin);

scanf("");

cvDestroyAllWindows();

/\*cvReleaseCapture(&camera);\*/

cvReleaseMemStorage(&storage);

cvReleaseImage(&small\_image);

cvReleaseImage(&small\_grey\_image);

cvReleaseImage(&edge\_image);

return 0;

}