Coding in C++

|  |
| --- |
|  |
|  | #include "opencv2/imgproc.hpp" |
|  | #include "opencv2/highgui.hpp" |
|  | #include <iostream> |
|  |  |
|  | using namespace cv; |
|  |  |
|  | //![variables] |
|  | Mat src, src\_gray; |
|  | Mat dst, detected\_edges; |
|  |  |
|  | int lowThreshold = 0; |
|  | const int max\_lowThreshold = 100; |
|  | const int ratio = 3; |
|  | const int kernel\_size = 3; |
|  | const char\* window\_name = "Edge Map"; |
|  | //![variables] |
|  |  |
|  | /\*\* |
|  | \* @function CannyThreshold |
|  | \* @brief Trackbar callback - Canny thresholds input with a ratio 1:3 |
|  | \*/ |
|  | static void CannyThreshold(int, void\*) |
|  | { |
|  | //![reduce\_noise] |
|  | /// Reduce noise with a kernel 3x3 |
|  | blur( src\_gray, detected\_edges, Size(3,3) ); |
|  | //![reduce\_noise] |
|  |  |
|  | //![canny] |
|  | /// Canny detector |
|  | Canny( detected\_edges, detected\_edges, lowThreshold, lowThreshold\*ratio, kernel\_size ); |
|  | //![canny] |
|  |  |
|  | /// Using Canny's output as a mask, we display our result |
|  | //![fill] |
|  | dst = Scalar::all(0); |
|  | //![fill] |
|  |  |
|  | //![copyto] |
|  | src.copyTo( dst, detected\_edges); |
|  | //![copyto] |
|  |  |
|  | //![display] |
|  | imshow( window\_name, dst ); |
|  | //![display] |
|  | } |
|  |  |
|  |  |
|  | /\*\* |
|  | \* @function main |
|  | \*/ |
|  | int main( int argc, char\*\* argv ) |
|  | { |
|  | //![load] |
|  | CommandLineParser parser( argc, argv, "{@input | fruits.jpg | input image}" ); |
|  | src = imread( samples::findFile( parser.get<String>( "@input" ) ), IMREAD\_COLOR ); // Load an image |
|  |  |
|  | if( src.empty() ) |
|  | { |
|  | std::cout << "Could not open or find the image!\n" << std::endl; |
|  | std::cout << "Usage: " << argv[0] << " <Input image>" << std::endl; |
|  | return -1; |
|  | } |
|  | //![load] |
|  |  |
|  | //![create\_mat] |
|  | /// Create a matrix of the same type and size as src (for dst) |
|  | dst.create( src.size(), src.type() ); |
|  | //![create\_mat] |
|  |  |
|  | //![convert\_to\_gray] |
|  | cvtColor( src, src\_gray, COLOR\_BGR2GRAY ); |
|  | //![convert\_to\_gray] |
|  |  |
|  | //![create\_window] |
|  | namedWindow( window\_name, WINDOW\_AUTOSIZE ); |
|  | //![create\_window] |
|  |  |
|  | //![create\_trackbar] |
|  | /// Create a Trackbar for user to enter threshold |
|  | createTrackbar( "Min Threshold:", window\_name, &lowThreshold, max\_lowThreshold, CannyThreshold ); |
|  | //![create\_trackbar] |
|  |  |
|  | /// Show the image |
|  | CannyThreshold(0, 0); |
|  |  |
|  | /// Wait until user exit program by pressing a key |
|  | waitKey(0); |
|  |  |
|  | return 0; |
|  | } |