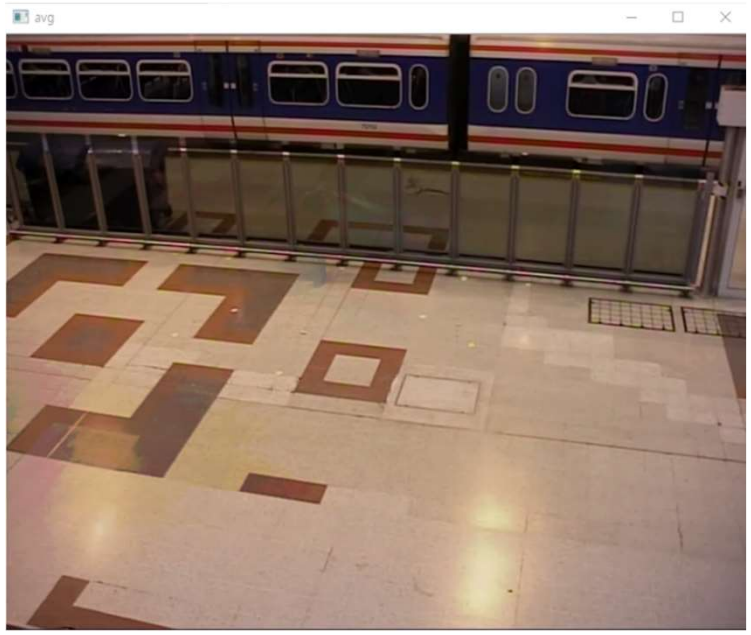
Background subtraction and Morphological operation

Sung Soo Hwang

Background subtraction

- Example code
  - Generating average image

```
int main() {
	VideoCapture capture("background.mp4");
	Mat image, sum, avg;
	int cnt = 2;

	capture >> avg;

	while (true) {
		if (!capture.read(image)) break;
		add(image / cnt, avg*(cnt - 1) / cnt, avg);

		imshow("avg", avg);
		cnt++;
		waitKey(33);
	}
}
```



Background subtraction

- Example code(using absdiff)

```
int main() {
	VideoCapture capture("background.mp4");
	Mat background, image, gray, result, foregroundMask, foregroundImg;

	//set the first frame as background
	capture >> background;
	cvtColor(background, background, CV_BGR2GRAY);

	while (true) {
		if (capture.grab() == 0) break;
		capture.retrieve(image);
		cvtColor(image, gray, CV_BGR2GRAY);

		absdiff(background, gray, foregroundMask);
		threshold(foregroundMask, foregroundMask, 50, 255, CV_THRESH_BINARY);
		foregroundMask.copyTo(foregroundImg);
		gray.copyTo(foregroundImg, foregroundMask);

		imshow("foregroundImg", foregroundImg);
		imshow("foregroundMask", foregroundMask);
		imshow("background", background);

		waitKey(33);
	}
}
```

Background subtraction

- Example code(using absdiff)

Background subtraction

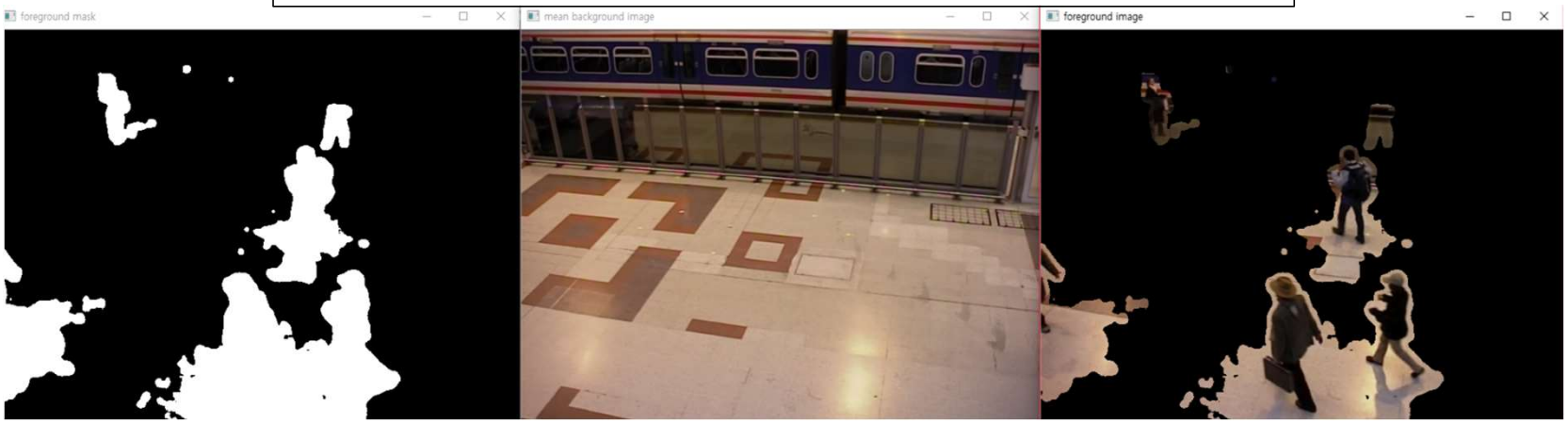
- Example code
  - openCV MoG2

```
int main() {
	Ptr<BackgroundSubtractor> bg_model = createBackgroundSubtractorMOG2();
	Mat image, foregroundMask, backgroundImg, foregroundImg;
	VideoCapture cap("background.mp4");

	while (true) {
		cap >> image;
		resize(image, image, Size(640, 480));

		if (foregroundMask.empty())
			foregroundMask.create(image.size(), image.type());

		// image: Next video frame. Floating point frame will be used without scaling and //should be in range [0,255].
		// foregroundMask: The output foreground mask as an 8-bit binary image.
		bg_model->apply(image, foregroundMask);
		GaussianBlur(foregroundMask, foregroundMask, Size(11, 11), 3.5, 3.5);
		threshold(foregroundMask, foregroundMask, 10, 255, THRESH_BINARY);
		foregroundImg = Scalar::all(0);
		image.copyTo(foregroundImg, foregroundMask);
		// backgroundImg: The output background image.
		bg_model->getBackgroundImage(backgroundImg);
```

Background subtraction

- Example code
  - openCV MoG2

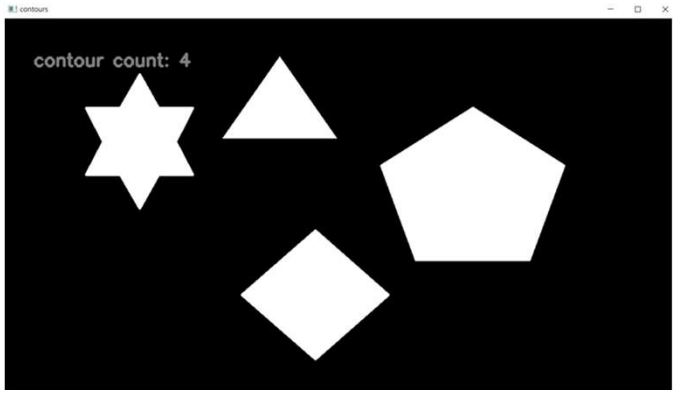
```
                imshow("foreground mask", foregroundMask);
                imshow("foreground image", foregroundImg);

                if (!backgroundImg.empty()) {
                        imshow("mean background image", backgroundImg);
                }
                waitKey(33);
        }
}
```



Counting the number of objects

- Example code

```
int main() {
	Mat gray = imread("contours.png", 0);
	Mat result;
	threshold(gray, result, 230, 255, THRESH_BINARY_INV);
	vector<vector<Point>> contours;
	vector<Vec4i>hierarchy;
	findContours(result, contours, hierarchy, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE);

	putText(result, format("contour count: %d", contours.size()), Point(50, 80), FONT_HERSHEY_SIMPLEX, 1, Scalar(12
	8), 4);

	imshow("contours", result);
	waitKey(0);
}
```



Counting the number of objects

- Retrival Modes

Enumerator	
RETR_EXTERNAL Python: cv.RETR_EXTERNAL	retrieves only the extreme outer contours. It sets `hierarchy[i][2]=hierarchy[i][3]=-1` for all the contours.
RETR_LIST Python: cv.RETR_LIST	retrieves all of the contours without establishing any hierarchical relationships.
RETR_CCOMP Python: cv.RETR_CCOMP	retrieves all of the contours and organizes them into a two-level hierarchy. At the top level, there are external boundaries of the components. At the second level, there are boundaries of the holes. If there is another contour inside a hole of a connected component, it is still put at the top level.
RETR_TREE Python: cv.RETR_TREE	retrieves all of the contours and reconstructs a full hierarchy of nested contours.
RETR_FLOODFILL Python: cv.RETR_FLOODFILL	

- Approximation modes

Enumerator	
CHAIN_APPROX_NONE Python: cv.CHAIN_APPROX_NONE	stores absolutely all the contour points. That is, any 2 subsequent points (x1,y1) and (x2,y2) of the contour will be either horizontal, vertical or diagonal neighbors, that is, max(abs(x1-x2),abs(y2-y1))==1.
CHAIN_APPROX_SIMPLE Python: cv.CHAIN_APPROX_SIMPLE	compresses horizontal, vertical, and diagonal segments and leaves only their end points. For example, an up-right rectangular contour is encoded with 4 points.
CHAIN_APPROX_TC89_L1 Python: cv.CHAIN_APPROX_TC89_L1	applies one of the flavors of the Teh-Chin chain approximation algorithm **[201]**
CHAIN_APPROX_TC89_KCOS Python: cv.CHAIN_APPROX_TC89_KCOS	applies one of the flavors of the Teh-Chin chain approximation algorithm **[201]**

Counting the number of objects

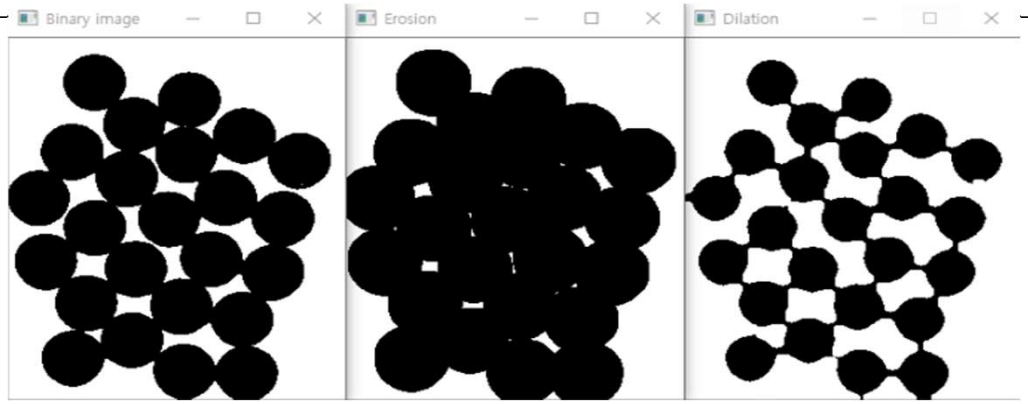
- Example code

```
int main() {
	Mat gray = imread("contours.png", 0);
	Mat result;
	threshold(gray, result, 180, 255, THRESH_BINARY_INV);
	vector<vector<Point>> contours;
	vector<Vec4i>hierarchy;
	findContours(result, contours, hierarchy, CV_RETR_EXTERNAL, CV_CHAIN_APPROX_SIMPLE);

	//defining bounding rectangle
	vector<Rect> boundRect(contours.size());
	for (int i = 0; i < contours.size(); i++)
		boundRect[i] = boundingRect(Mat(contours[i]));
	//draw rectangles on the contours
	for (int i = 0; i < contours.size(); i++)
		rectangle(result, boundRect[i].tl(), boundRect[i].br(), Scalar(0, 0, 255), 2, 8, 0);
	imshow("contours", result);
	waitKey(0);
}
```

Erosion and dilation

- Example code

```
Mat image, erosion, dilation;
Mat element = getStructuringElement(MORPH_ELLIPSE, Size(10, 10));
//Other options:MORPH_RECT, MORPH_CROSS

image = imread("water_coins.jpg", 0);
threshold(image, image, 128, 255, THRESH_BINARY);

erode(image, erosion, element);
dilate(image, dilation, element);

imshow("Binary image", image);
imshow("Erosion", erosion);
imshow("Dilation", dilation);
waitKey(0);
```



Erosion and dilation

- Example code

```
Mat image, opening, closing, element;
image = imread("water_coins.jpg", 0);

threshold(image, image, 128, 255, THRESH_BINARY);
element = getStructuringElement(MORPH_ELLIPSE, Size(7, 7));

morphologyEx(image, closing, MORPH_CLOSE, element);
morphologyEx(image, opening, MORPH_OPEN, element);

imshow("Binary image", image);
imshow("opening", opening);
imshow("closing", closing);
waitKey(0);
```