void featureDetection(Mat img\_1, vector<Point2f>& points1) {

vector<KeyPoint> keypoints\_1;

int fast\_threshold = 20;

bool nonmaxSuppression = true;

FAST(img\_1, keypoints\_1, fast\_threshold, nonmaxSuppression);

KeyPoint::convert(keypoints\_1, points1, vector<int>());

}

void featureTracking(Mat img\_1, Mat img\_2, vector<Point2f>& points1, vector<Point2f>& points2, vector<uchar>& status) {

vector<float> err;

Size winSize=Size(21,21);

TermCriteria termcrit=TermCriteria(TermCriteria::COUNT+TermCriteria::EPS, 30, 0.01);

calcOpticalFlowPyrLK(img\_1, img\_2, points1, points2, status, err, winSize, 3, termcrit, 0, 0.001);

int indexCorrection = 0;

for( int i=0; i<status.size(); i++)

{ Point2f pt = points2.at(i- indexCorrection);

if ((status.at(i) == 0)||(pt.x<0)||(pt.y<0)) {

if((pt.x<0)||(pt.y<0)) {

status.at(i) = 0;

}

points1.erase (points1.begin() + i - indexCorrection);

points2.erase (points2.begin() + i - indexCorrection);

indexCorrection++;

}

}

}

E = findEssentialMat(points2, points1, focal, pp, RANSAC, 0.999, 1.0, mask);

recoverPose(E, points2, points1, R, t, focal, pp, mask);