Ignore. Import content3 from here.

use author=string

2018-05-27

Abstract

Introduction

Green Screens, are used in movies, series, news, video-games and home-made videos with the main reason of providing an artificial background. The technique requires an approximately monochrome and plain background (often green) to be placed behind whatever is wanted in the foreground (often a person). When the green screen is placed, a technique called Chroma Keying is applied in order to map where the plain green surface is. When the location of the green pixels are knows, a CGI, Computer Generated Image, is mapped onto these pixels, making it seem like the person is standing in front of whatever is put in the back.

Our algorithm provides a Green Screen tool for people, without having to place a Green Screen behind them. Our algorithm requires only 1 camera and has no dependence on color segmentation. Even though Computer Vision has taught us powerful, color based segmentation methods, they rely on plain backgrounds. Colour based segmentation is therefore not in our interest, as we seek to create something more flexible. Instead of using gaussian classification on the local colorspace to extract people from the image, we propose a method relying on feature detection, contouring and laplacian blurring, as an alternative.

Idea Process, 2D/3D, and Library Independence:

In the beginning we had an idea to try to go into segmentation and explore methods to possibly segment people in 3D. The concept there was to try to find the enveloping hull that closest fit the human being in 3D, using only a single camera and known geometric shapes that approximate the human form.

In order to do that we looked into using Pose-estimators such as OpenPose, and generating Spheres, Cubes, etc. to envelop the person.

On further feedback, we were guided towards doing more image segmentation, and less of the first concept to "scan" a human being into a 3D model. It was even suggested that we would/could try to map the images as we captured them onto a 3D model.

However, in order to do so, we would need to generate a 3D imaging tool which would allow us to both segment, capture, create a mosaic, and use something like ORBSLAM to keep track of the features and keypoints on this expansive surface that covers a human being.

Working on how to implement this, we decided therefore to first get the 2D elements correct. That is; To do segmentation using geometric shapes (Triangles, Circles, Polygons, etc.) and then, if time allowed us, to use these shapes as projected images of the 3D shapes of pyramids, Spheres, etc.

Along the way we would then also need to be able to extract the portions of the image that correlates to the overlap of these shapes and the camera feed. Which essentially turns it into a green-screen (without the green).

Ideally we would have a code that can take any coupled set of Canonical shapes, and a Pose-detection software, map the Shapes to the Pose, and from there be able to extract the information necessary.

The concept should hold for any kind of object, be it a simple or complicated one ideally.

The Process.:

Along the long road towards a usable 2D segmentation we tried a few different approaches.

First out was to use a Canny Edge detection algorithm and try to filter out which edges were closest to any pose-points. After exhausting our paths that way, we determined that it would be better to operate on

just a portion of the image. So we rewrote the algorithm to extract regions around points of interest, using element-wise multiplication with Filters.

These Filters were geometric shapes, and we first tried to simply extract the portions of the Canny image that were along the border of our picture. With the goal of using this then with the convexHull-function in openCV.

Our next attempt brought in a bit more interesting twists, as we started playing around with Laplacian Pyramids, and the concept of using the lower tiers of the Laplace Pyramid as our edge-detector. (The concept is not unlike extracting Detail-Spaces using Wavelets)

This turns out to maybe have some uses, but we are still then on a more or less plain segmentation-implementation.