**Posing and Projecting Faces**

Whew, we isolated the faces in our image. But now we have to deal with the problem that faces turned different directions look totally different to a computer:

A person taking a selfie

Description automatically generatedA person wearing a blue hat

Description automatically generatedA person wearing a hat and glasses

Description automatically generated

Humans can easily recognize that both images are of Will Ferrell, but computers would see these pictures as two completely different people.

To account for this, we will try to warp each picture so that the eyes and lips are always in the sample place in the image. This will make it a lot easier for us to compare faces in the next steps.

To do this, we are going to use an algorithm called **face landmark estimation**. There are lots of ways to do this, but we are going to use the approach [invented in 2014 by Vahid Kazemi and Josephine Sullivan.](http://www.csc.kth.se/~vahidk/papers/KazemiCVPR14.pdf)

The basic idea is we will come up with 68 specific points (called *landmarks*) that exist on every face — the top of the chin, the outside edge of each eye, the inner edge of each eyebrow, etc. Then we will train a machine learning algorithm to be able to find these 68 specific points on any face:

A dot to dot game

Description automatically generated

The 68 landmarks we will locate on every face. This image was created by [Brandon Amos](http://bamos.github.io/) of CMU who works on [OpenFace](https://github.com/cmusatyalab/openface" \t "_blank).

Here’s the result of locating the 68 face landmarks on our test image:

A person taking a selfie

Description automatically generatedA person wearing a hat and glasses

Description automatically generated

**PROTIP**: You can also use this same technique to implement your own version of Snapchat’s real-time 3d face filters!

Now that we know were the eyes and mouth are, we’ll simply rotate, scale and [shear](https://en.wikipedia.org/wiki/Shear_mapping#/media/File:VerticalShear_m%3D1.25.svg) the image so that the eyes and mouth are centered as best as possible. We won’t do any fancy 3d warps because that would introduce distortions into the image. We are only going to use basic image transformations like rotation and scale that preserve parallel lines (called [affine transformations](https://en.wikipedia.org/wiki/Affine_transformation)):

Now no matter how the face is turned, we are able to center the eyes and mouth are in roughly the same position in the image. This will make our next step a lot more accurate.