## Canny Count ITH

After performing Canny on the image, count the number of non-zero pixels. Do this along a range of thresholds.

(See messages I sent to Michelle today to see full algorithm details)

Started writing the ITH Evalator to see how well ITH’s perform. Are they too general? Etc.

# Messages to Michelle

So you know already that my project is this algorithm that identifies images as similar. The way it does that the steps of the algorithm are: For each image, For each ITH (will explain "ITH" in a sec) Assign one or more trait scores, with a margin of error for each. Once this has been done for all images, group images together whose traits scores are all within the margin of error for each other.

##### **Jordan**

So "an algorithm" is basically a series of steps you take to do something. So that little description up there is my "algorithm", and it's up to me to program it.

ITH stands for Image Trait Handler. Generally speaking, a trait is an aspect of an object that is consistent across all images of said object.

For example, for a Domino's logo, you could look at the colour frequency, the shapes, etc. It can be argued that once you define enough traits that make an image unique, if two images share all of those traits then they are likely the same object.

So I've coded the algorithm in a general sense: my program follows all steps and groups stuff together. But my mission now is to add more and more trait handlers that look at one trait in particular (such as the color) and provides a score accordingly.

Because right now, with the Image Trait Handlers I have, I get a lot of false positives: results that say two things are similar, when they actually aren't.

##### **Jordan**

I only have four ITHs at the moment, and things are coming along well. Those are: Color Frequency, Image Dimensions, Foreground detection, and Noise Detection.

For example, we can look at Noise Detection, which is the name I've given to the trait that describes how often pixels suddenly change in color. For example...

For this image:

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15978365_10212204707775582_636314610_n.jpg?oh=2c75f4fd97863f5d0e77b40955815ff7&oe=58781BAA

I perform the "Canny" algorithm to detect edges within the image:

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15978299_10212204709855634_230661622_n.jpg?oh=1d93caa50a07035d7dce4f8970b7a9ac&oe=58783AD8

I can adjust a slider of sorts to change its sensitivity to edges. If I want it more sensitive, I get something like this:

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15978481_10212204713495725_108516821_n.jpg?oh=7d96a3c7370c90eab4a8ce7ac62d43e4&oe=58794CDD

Granted, it's a lot noiser now, but that doesn't matter. What's important is that from this, you can see there is a lot of white consistently found throughout the image. This is true for a lot of images, but particularly consistent for any other images of that card

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15969834_10212204721975937_1762806744_n.jpg?oh=869203b964cda82a8f7ad0fb473988aa&oe=58781D15

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15970444_10212204722055939_966717985_n.jpg?oh=627d67e5c408f232e4a0fe3fb8348be4&oe=587864B7

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15978525_10212204722095940_958899121_n.jpg?oh=df8c21dc0992ac264d4617c9bb2490e2&oe=58793A11

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15942264_10212204722535951_1176824479_n.jpg?oh=f36c470eddea1866012ed58e6e294480&oe=587823DA

These are the results for different images of the same card.

Here is the result for a completely different image:

https://scontent-lht6-1.xx.fbcdn.net/v/t34.0-12/15978867_10212204726216043_1703668306_n.jpg?oh=f256d95773a11bb5910ef7c9fc051337&oe=58783151

(That's a speed bump sign :P)

Naturally, as a human, you can tell these are incredibly different. The first images were a bunch of static throughout the bounds of the image. The second one is a collection of smooth lines with big chunks of black within and around them.

A computer can break this down in a simpler way. I don't need to emulate human intuition, I just need an algorithm that comes to the same general conclusion

And it comes down to this:

1. Break the image up into 3 rows and 3 columns, making 9 cells. 2. For each cell, count the number of white pixels. 3. Assign a margin of error for any given cell.

I can say that, for images of the christmas card, every single cell contains roughly 300 white pixels.

However, for images of the triangular sign, the top left and top right cells have almost no white pixels.

While the other cells have roughly 100 white pixels each.

And thus, I have defined an algorithm to analyse a particular trait of the image.

This is our new Image Trait Handler for today!

Now, obviously, these single ITH isn't enough on its own. Lots of complicated christmas cards will have the same noise frequency pattern: ~300 white pixels in each cell in the 3x3 grid. And pretty much any triangular sign will have no white pixels in the top right and top left.

##### **Jordan**

But by adding this ITH to the algorithm as a whole, it acts as an adviser of sorts who tells the main algorithm, "Bro, I handle the noise frequency distribution. Image A and Image B look similar as far as I'm concerned." They might be right, they might be wrong, but if the Advisor of Color is all like, "Mate, they've got the same colors too." and the Advisor of Blah Blah Blah says, "I'm happy with it too." then all of the ITH's independently think the images are similar, so we increase our chances.

##### **Jordan**

That's why it's my mission to add more and more ITH's. If we get enough advisors giving the thumbs up, like a million advisors or something, looking at a wide enough range of traits, then that probability of a true positive match approaches 100%

Of course, I'm not going to add a million Image Trait Handlers that will do this for me. At most, I'll probably do somewhere between 10 and 20, depending on how effective they are

##### **Jordan**

That's why I need 100% true positives for each ITH, but any number of false positives is fine, where a positive result is one that says two images are similar. It's a true positive if they actually are similar, and it's a false positive if it says they're similar, but they're not.

##### **Jordan**

A 100% true positive rate is easy. If each ITH is general enough, and offers a reasonable margin of error, true positive matches are easy to come by. The difficulty is keeping this 100% rate while still be critical enough to weed out the false matches.

I'm happy if a noisy, green christmas card is matches with a noisy red christmas card. They're both noisy, that's fair enough, and I can define them by different by adding an ITH saying they're different because they are different colors.

Anyway, things are probably starting to get a bit confusing 😕

Just scrolled up and realized I've said loooooads today, so I'll leave it there haha.

But hopefully that gives you a good idea of what I'm trying to do here, big picture!