Professor Anil Kokaram

Anil.Kokaram@tcd.ie

5C1 Motion Picture Engineering

Digital Video Engineering

Engineering for Moving Pictures

5C1 : Motion Picture Engineering

- Pixel Pushing (The invisible tools for making better pictures)
- Pixel Communications (Video Streaming)
- Pixel Quality Measurement

		9	10	11	12	13	14	15	16	17
Monday	Lecture	EEP55M08 - M17	EEMT17 - MMTLAB	EEMT17 - MMTLAB	EE5C01 M21	EEP55M08 - PBLT	EE5M02 - SPSR CS7NS6 - 1.07 LLOYD	EE5M02 - SPSR CS7NS6 - 1.07 LLOYD	EE5C01 - DO CS7NS5 - LB01	CS7NS5 - LB01 EE5M01 - M20
	Tutorial									
	Lab									
Tuesday	Lecture	EEP55M08 - SYNGE	EEP55C25 - CEDR CS7NS6 - 1.07 LLOYD	EEP55M08 - AP2.28 CS7GV5 - 4.09 LEINSTER		CS7GV3 - 1.20 LLOYD	CS7GV3 - 1.20 LLOYD EE5M02 - AP2.02	CS7DS2 - LB04 EE5C01 - AP2.28	EE5C01 - AP2.28	EE5M01 - AP2.28
	Tutorial									
	Lab									
Wednesday	Lecture	EEP55C24 - AP2.04	EEP55C24 - AP2.04	CS7GV5 - 4.09	EE5M02 - AP2.02		CS7NS2 - 1.07 LLOYD	CS7DS2 - LB08	ME5E3 - PBENG	ME5E3 - PBENG CS7NS5 - LB04
	Tutorial									
	Lab									
Thursday	Lecture		CS7IS2 - LB01		EEP55C24 - AP2.04	EEP55C24 - M17	EEP55C23 - PBENG	EEP55C23 - PBENG CS7CS3 - LB01	EEP55C25 - AP2.28 CS7CS3 - LB01	EEP55C25 - AP2.28 CS7CS3 - LB01
	Tutorial									
	Lab									
Friday	Lecture	EE5M01 AP2.28	EE5M01 - AP2.28	EE5C01 - AP2.28 CS7NS6 - 1.07 LLOYD	EE5C01 - AP2.28	CS7NS2 - 1.07 LLOYD	CS7IS2 - LB08			MECHLB
	Tutorial									
	Lab									

LECTURES/Demos

NUKE (weeks 1,2,3), Assignment time weeks 4-6
Compression Lab week 6, Assigment time weeks 9-12

ClassTests/Industry Talks

Some speakers in CA and NZ
so we'll have to get creative

Course Content

- Week 1 Introduction + Colour Segmentation
- Week 2 Perception and Bayesian Inference for Colour Segmentation
- Week 3 Video Processing and Motion Estimation
- Week 4 Deep learning and motion estimation
- Week 5 Applications in Video Processing
- Week 6 Another introduction to compression
- Week 8-12 Modern Video Compression and the Engineering behind YouTube and Netflix

Practical Work

- Week 1+2 : NUKE for Compositing
- Week 3 : Scripts in NUKE for video processing
- Week 4 – 6 : Writing your own script/graph for colour keying in video and writing a 4 page report
- Week 8 – 12 : Video Transcoding and Quality Measurement

Course Texts and Online Resources

Texts

- **Video Processing, Murat Tekalp**
- Markov Random Fields for Vision and Image Processing. Edited by A. Blake, P. Kohli and C. Rother, MIT Press, 2011. ISBN: 978-0-262-01577-6
- The Essential Guide to Video Processing. A. Bovik, Academic Press, 2009. ISBN: 978-0-12-374456-2

Online Resources

- IEEE Explore
- Various papers I will point out
- YouTube links as we go
- Shapiro’s course on “Mars to Hollywood” and Katsagellos “Video and Image Processing” are references

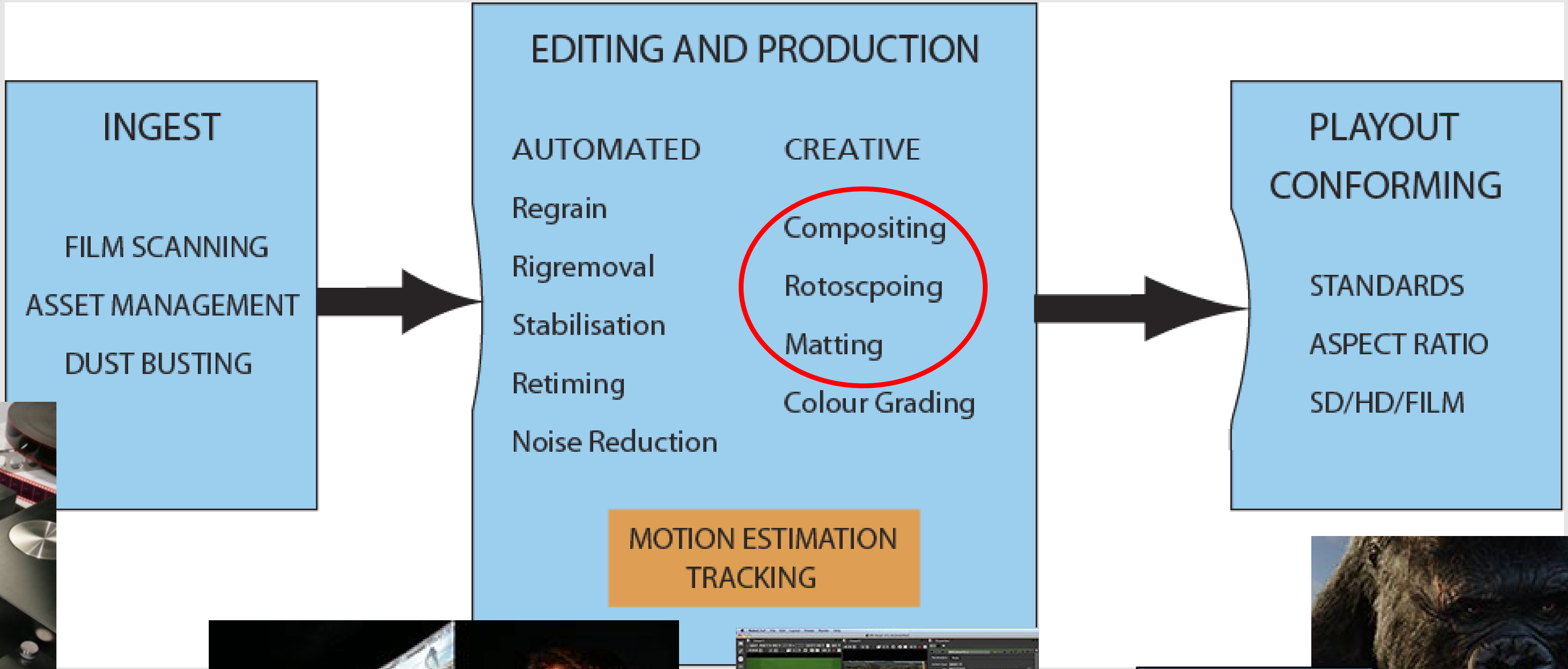
The technical material discussed in this course is at the bleeding edge of what is possible. Get comfortable with not being able to find a complete set of notes like you’d find in a book which you can then just learn off. The book is still being written.

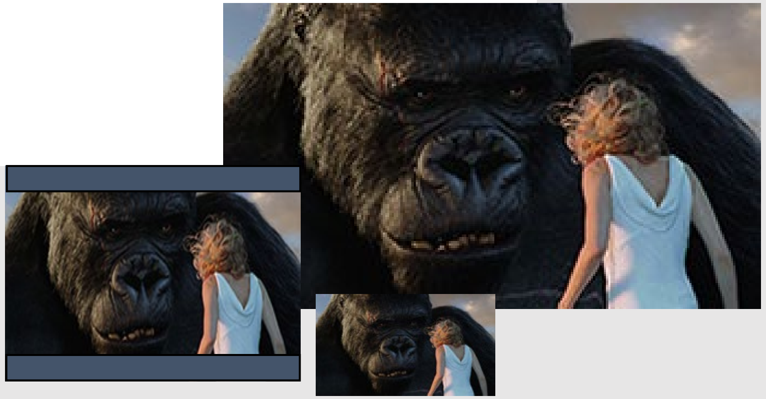
NUKE®

Powerful compositing, editorial and review.

Download trial | Buy

The Post Production Workflow



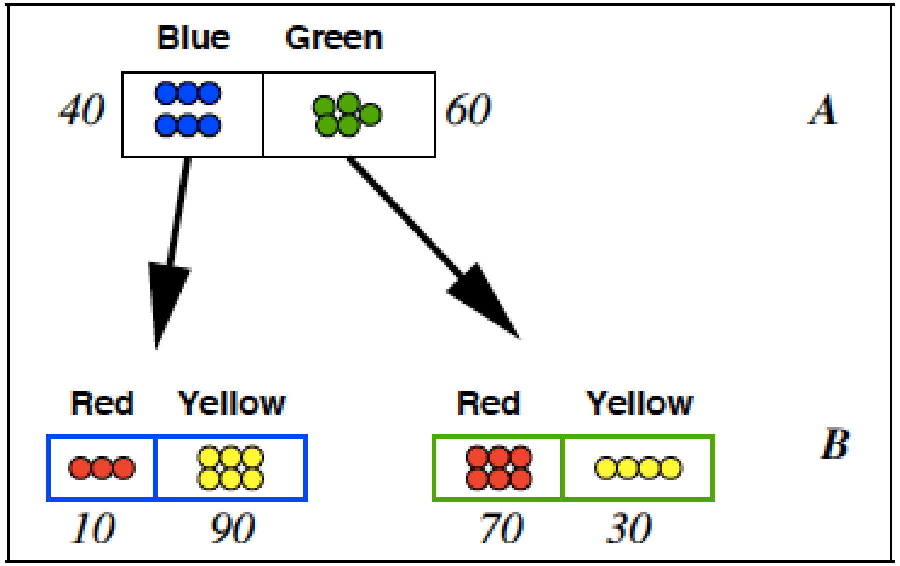


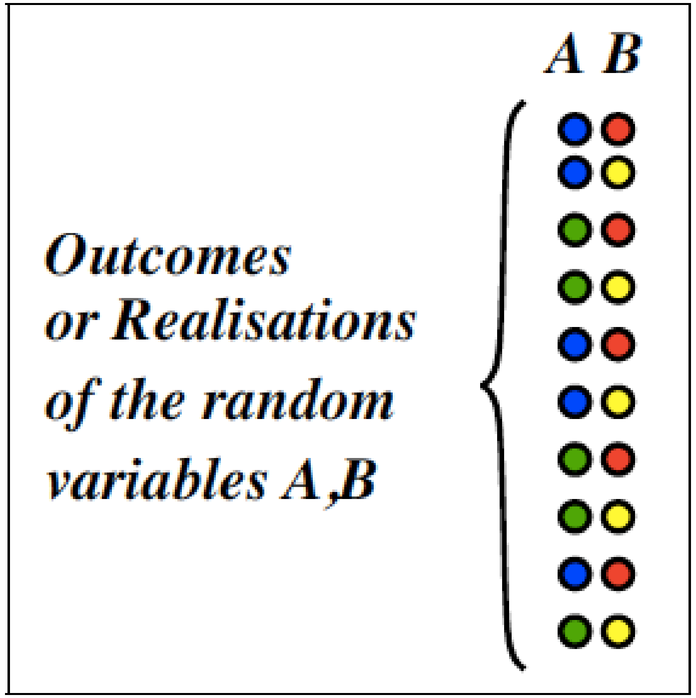
A short review of Bayesian Inference

The best way to think about a problem (in my opinion)

- Using probability allows you to explicitly capture your uncertainty about the world
- Well established in Signal processing since the 1980’s. Very well known in Statistics since forever. (Rev Bayes).
- Every well known framework has a Bayesian interpretation including Machine Learning
- Leads to “Energy minimisation” naturally
- But don’t get too religious about it.

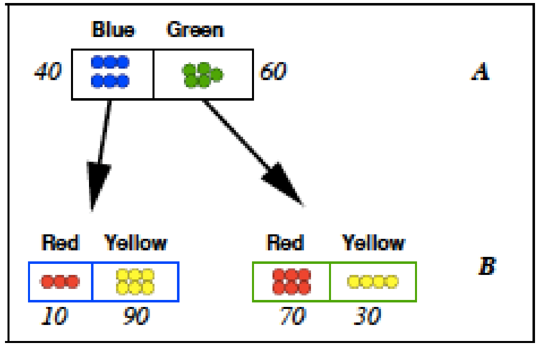
Probability and marbles





- Consider Two random variables A, B, which each have two outcomes or two realisations *(blue/green)*, *(red/yellow)* respectively
- Outcomes are generated by first selecting a marble from box A, then selecting a marble from a box B. But box B depends on what colour you select from box A.

Probability and marbles



- Probability is "just a number". A probability density function expresses the probability of all outcomes of a R.V. and must obey the following equation.

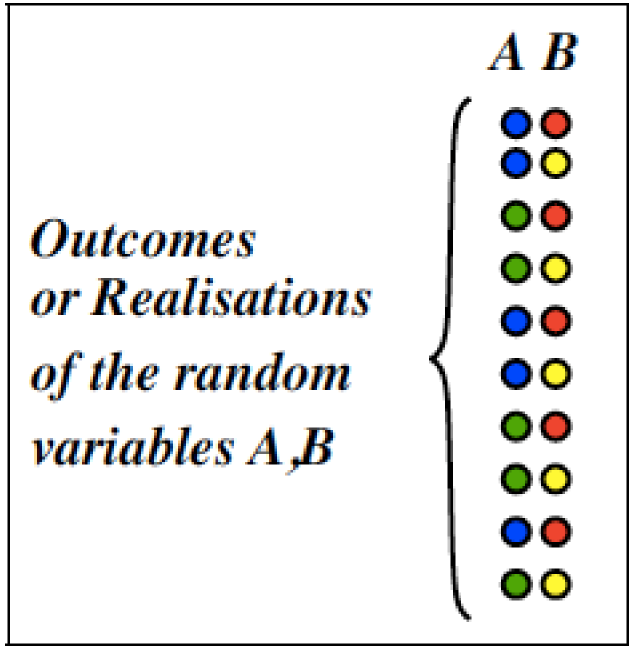
$$\int p(A)dA = 1$$

- In our example

$$\int p(A)dA = p(A = blue) + p(A = green) = \frac{40}{100} + \frac{60}{100} = 1.0$$

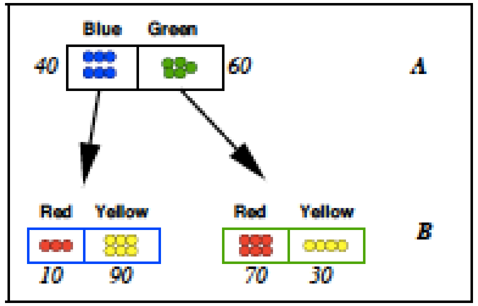
- What is $p(B = red|A = blue)$? The probability of *realising* or *observing* a red marble from r.v. B **GIVEN** that a blue marble was drawn from the probability distribution for r.v. A (i.e. a blue marble was observed or realised as the outcome from r.v. A).
- $p(B = red|A = blue)$ is the Conditional probability of observing a red marble from box B **GIVEN** a particular outcome from A. It is the probability of B conditioned on A.
- $p(B = red, A = blue)$ is the **JOINT** probability of observing a red marble from B **AND** a blue marble from A
- $p(B, A)$ is the **JOINT** probability distribution of A **AND** B

Probability and marbles



- We can measure $p(B = red|A = blue)$, or $p(B = red, A = blue)$, by *simulating* the system (with Matlab say) and observing the frequencies of the various outcomes.
- In this example of a set of simulated outcomes, $p(B = red|A = blue) = 3/5$, $p(B = red, A = blue) = 3/10$.
- We would need *lots* of example outcomes to be sure of our *measurements*.
- But we can calculate these values using laws of probability

Probability and marbles



$$p(B = red|A = blue) = 0.1 \text{ Just by reading it off from our model}$$

$$p(B = red, A = blue) = p(B = red|A = blue)p(A = blue) = 0.1 \times 0.4 = 0.04$$

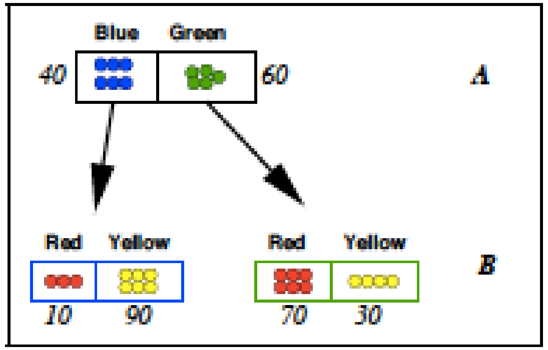
$$p(B = red|A = blue) = \frac{p(B = red, A = blue)}{p(A = blue)} = 0.04/.4 = 0.1$$

This is an important equation in conditional probability

$$p(B|A) = \frac{p(B, A)}{p(A)} = \frac{p(A, B)}{p(A)}$$

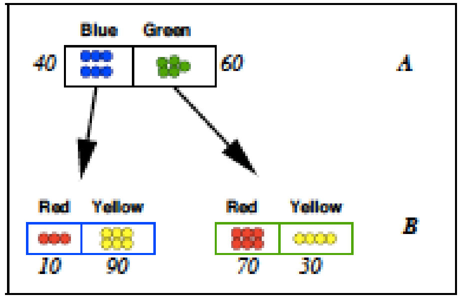
$$p(A|B) = \frac{p(A, B)}{p(B)} = \frac{p(B, A)}{p(B)}$$

Probability and marbles



- Bayes' theorem turns out to be 'just' another law of conditional probability
- What is $p(A = blue|B = red)$? Hmmm.... we can't just read that off easily now eh? Its sort of upside down ...
- Given that you have observed that $B = red$ what is the probability that a blue marble was drawn from A? This is the kind of thing that you end up having to answer alot in signal processing. Given that the corrupted speech signal at this point is 0.6Volts, what is the probability that the actual signal is 1.0 Volts?

Probability and marbles



$$p(A = blue|B = red) = \frac{p(B = red, A = blue)}{p(B = red)} = \frac{p(B = red|A = blue)p(A = blue)}{p(B = red)}$$

We can find $p(B = red|A = blue)$ easily now

$$= \frac{0.1 \times 0.4}{p(B = red)}$$

(10 + 70)/(200)

This is Bayes' Theorem

$$p(B|A) = \frac{p(A|B)p(B)}{p(A)} \tag{1}$$

It turns the potentially tricky problem of estimating $p(B|A)$ into the hopefully easier problem of estimating $P(A|B)$.

Probability and marbles

This is Bayes' Theorem

$$p(B|A) = \frac{p(A|B)p(B)}{p(A)} \tag{2}$$

- $p(A|B)$ is the **Likelihood**.
- $p(B)$ is the **Prior**.
- $p(A)$ is the normalising factor or sometimes used as **Evidence**.
- $p(B|A)$ is the **Posterior** distribution because you are asking questions *after the fact*.

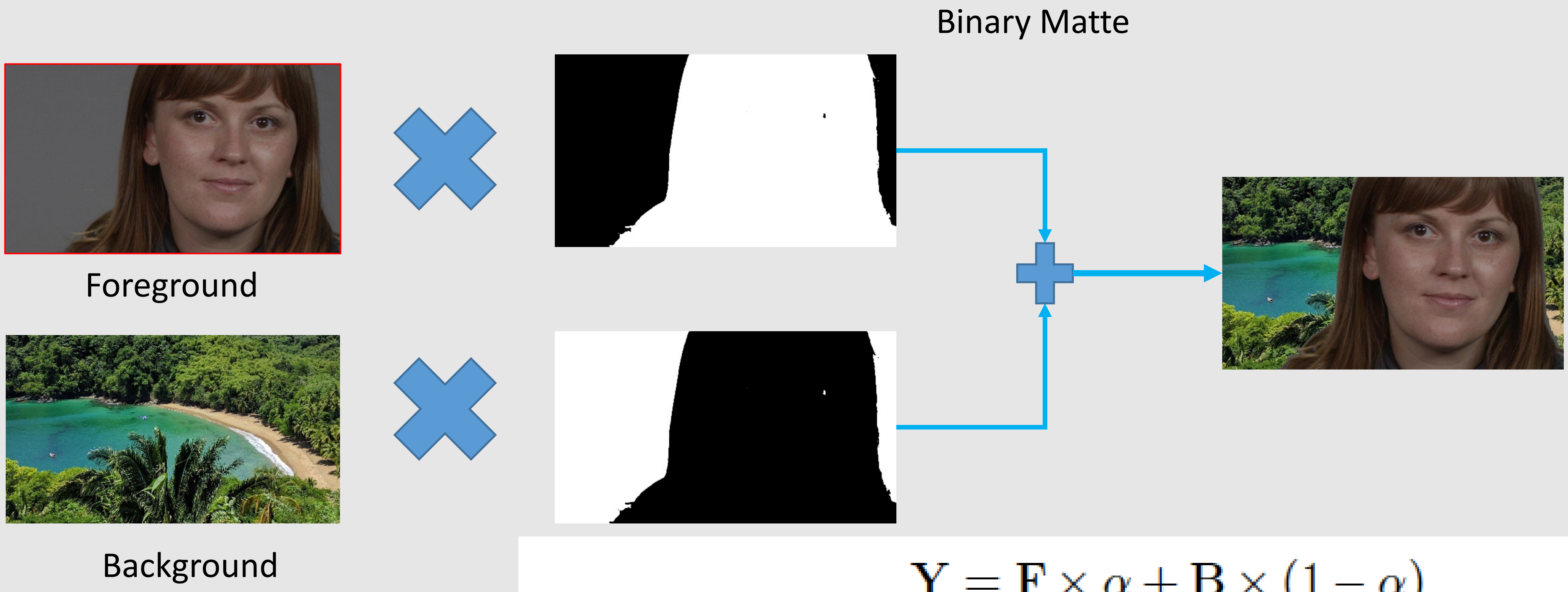
Greenscreen for compositing

Greenscreen for compositing

Actually a “grey” screen in this case

$\mathbf{x}$ is a position i.e. $\mathbf{x} = [h, k]$ where h is the column position of a pixel and k is the row position.

Compositing



$$\mathbf{Y} = \mathbf{F} \times \alpha + \mathbf{B} \times (1 - \alpha)$$

$\mathbf{x}$ is the position of the pixel with value $I(\mathbf{x})$ Let $\alpha(\mathbf{x})$ be the binary matte value at that site. Foreground is indicated by $\alpha = 1$

Colour Keying

$$\mathbf{Y} = \mathbf{F} \times \alpha + \mathbf{B} \times (1 - \alpha)$$

$\mathbf{x}$ is the position of the pixel with value $I(\mathbf{x})$ Let $\alpha(\mathbf{x})$ be the binary matte value at that site. Foreground is indicated by $\alpha = 1$

Binary Matte

Foreground

Background

Colour segmentation/keying is about discovering $\alpha(\mathbf{x})$ based primarily on colour alone.

Let us consider $p(\alpha(\mathbf{x}|I(\mathbf{x}))$

We want to choose α which is most probable GIVEN the observed value of a pixel $I(\mathbf{x})$ at that same site.

If $p(\alpha = 1|I) > p(\alpha = 0|I)$ then $\alpha = 1$ and that site must be foreground

If $p(\alpha = 0|I) > p(\alpha = 1|I)$ then $\alpha = 0$ and that site must be background

The implicit algorithm here is that we are making decisions on a per-pixel basis. Visting each site in turn asking "Does this pixel colour look more like the background colour than the foreground colour?" If it does then it must be the background then.

Likelihood Prior

$$p(\alpha(\mathbf{x})|I(\mathbf{x})) = \frac{p(I(\mathbf{x})|\alpha(\mathbf{x}))p(\alpha)}{p(I(\cdot))}$$

Evidence

$$p(\alpha|I) = \begin{cases} \frac{p(I|\alpha=1)p(\alpha=1)}{p(I(\cdot))} & \alpha = 1 \\ \frac{p(I|\alpha=0)p(\alpha=0)}{p(I(\cdot))} & \alpha = 0 \end{cases}$$

$$p(\alpha|I) = \begin{cases} \frac{p(I|\alpha=1)p(\alpha=1)}{p(I(\cdot))} & \alpha = 1 \\ \frac{p(I|\alpha=0)p(\alpha=0)}{p(I(\cdot))} & \alpha = 0 \end{cases}$$

Assuming we know nothing about the matte itself, the prior is a constant
Evidence doesn't change depending on \alpha so that can be ignored

$$p(\alpha|I) = \begin{cases} p(I|\alpha = 1) & \alpha = 1 \\ p(I|\alpha = 0) & \alpha = 0 \end{cases}$$

$$= \begin{cases} p_f(I = \text{Foreground Colour}) & \alpha = 1 \\ p_b(I = \text{Background Colour}) & \alpha = 0 \end{cases}$$

What is p(I=F) , p(I=B) ?

If I, the image pixels were just grey scale then we could say

$$p_f(I|\alpha = 1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \left[\frac{(I - \bar{F})^2}{2\sigma^2}\right]$$

where $\bar{F}$ is the *average* colour of the foreground.

But the whole point of this is COLOUR keying so…

For colour, the image pixel is actually a 3 component vector $\mathbf{I}(\mathbf{x}) = [I_r, I_g, I_b]$. So we need to instead use a 3-channel Gaussian distribution as follows for the Foregound.

$$p_f(I|\alpha = 1) \propto \exp - \left[\frac{(I_r - \bar{F}_r)^2}{2\sigma_r^2} + \frac{(I_g - \bar{F}_g)^2}{2\sigma_g^2} + \frac{(I_b - \bar{F}_b)^2}{2\sigma_b^2} \right]$$

where the average colour of the foreground is given by 3 colour means $\bar{F}_r, \bar{F}_g, \bar{F}_b$ for red green and blue respectively. This assumes that there is no inter-channel correlation.

We know r, g, b colourspace is not a good idea because in fact there is alot of correlation between the channels in that colourspace. So we'll use YUV (Luminance/Chrominance). There is a linear relationship between RGB and YUV but in YUV space much of the inter-channel correlation is gone.

R(ed)

G(reen)

B(lue)

$$\begin{bmatrix} Y \\ U \\ V \end{bmatrix} = \mathbf{C} \begin{bmatrix} R \\ G \\ B \end{bmatrix}$$

$$\text{Where} \quad \mathbf{C} = \begin{bmatrix} 0.3 & 0.6 & 0.1 \\ -0.15 & -0.3 & 0.45 \\ 0.4375 & -0.3750 & -0.0625 \end{bmatrix}$$

Y

U

V

[redacted]

So now we represnt an image pixel in YUV space so $\mathbf{I}(\mathbf{x}) = [I_y, I_u, I_v]$.

$$p_f(\mathbf{I}|\alpha = 1) \propto \exp - \left[\frac{(I_y - \bar{F_y})^2}{2\sigma_y^2} + \frac{(I_u - \bar{F_u})^2}{2\sigma_u^2} + \frac{(I_v - \bar{F_v})^2}{2\sigma_v^2} \right]$$

With foreground mean colour $[F_y, F_u, F_v]$

$$p_b(\mathbf{I}|\alpha = 0) \propto \exp - \left[\frac{(I_y - \bar{B_y})^2}{2\sigma_y^2} + \frac{(I_u - \bar{B_u})^2}{2\sigma_u^2} + \frac{(I_v - \bar{B_v})^2}{2\sigma_v^2} \right]$$

With background mean colour $[B_y, B_u, B_v]$

NOTE THAT WE ARE ASSUMING HERE THAT THE VARIANCES OF THE TWO COLOUR DISTRIBUTIONS ARE THE SAME!
IN REAL LIFE WE WON'T EXPECT THEM TO BE!

But $p(F)$ messy i.e. the foreground is most likely a mixture of of colours, not just one colour. So it can't be well modelled by a single Gaussian. However, the background is simple in fact we can make it be simple (hang a green cloth or use a big LED screen). So $p(B)$ can be engineered to be well modeled by a Gaussian.

That means in fact we can change our decision process to a 1 - class process. We simply check to see whether a pixel colour is likely to be background colour or not. We don't check to see whether its closer to the foreground than the background. We just stick our thumbs in the air and say well, let's measure the probability of it being background. If that's high $> k$ we say grand its background then.

If $p(\alpha = 0|I) > k$ then $\alpha = 0$ and that site must be background

How do we test $p(\alpha = 0|I) > k$?

$$p(\alpha = 0|I) = p_b(I = \text{background colour})$$

$$\Rightarrow p_b(I = \text{background colour}) \propto \exp - \left[\frac{(I_y - B_y)^2}{2\sigma_y^2} + \frac{(I_u - B_u)^2}{2\sigma_u^2} + \frac{(I_v - B_v)^2}{2\sigma_v^2}\right]$$

$$\Rightarrow \exp - \left[\frac{(I_y - B_y)^2}{2\sigma_y^2} + \frac{(I_u - B_u)^2}{2\sigma_u^2} + \frac{(I_v - B_v)^2}{2\sigma_v^2}\right] > k$$

Taking exponents is unstable in a computer so use log

$$\Rightarrow \left[\frac{(I_y - B_y)^2}{2\sigma_y^2} + \frac{(I_u - \bar{B}_u)^2}{2\sigma_u^2} + \frac{(I_v - \bar{B}_v)^2}{2\sigma_v^2}\right] < \ln(k)$$

Note the change in sign causes the $>$ to turn into a $<$. k is just a number .. a threshold.... so $\ln(k)$ is just some other threshold.

Algorithm

1. Measure Gaussian parameters for background somewhere in the example image. Get the mean and variance of each colour component in the background $\bar{B}_y, \sigma_y^2$, etc.

2. Set some threshold E_t for deciding whether a site is background or not

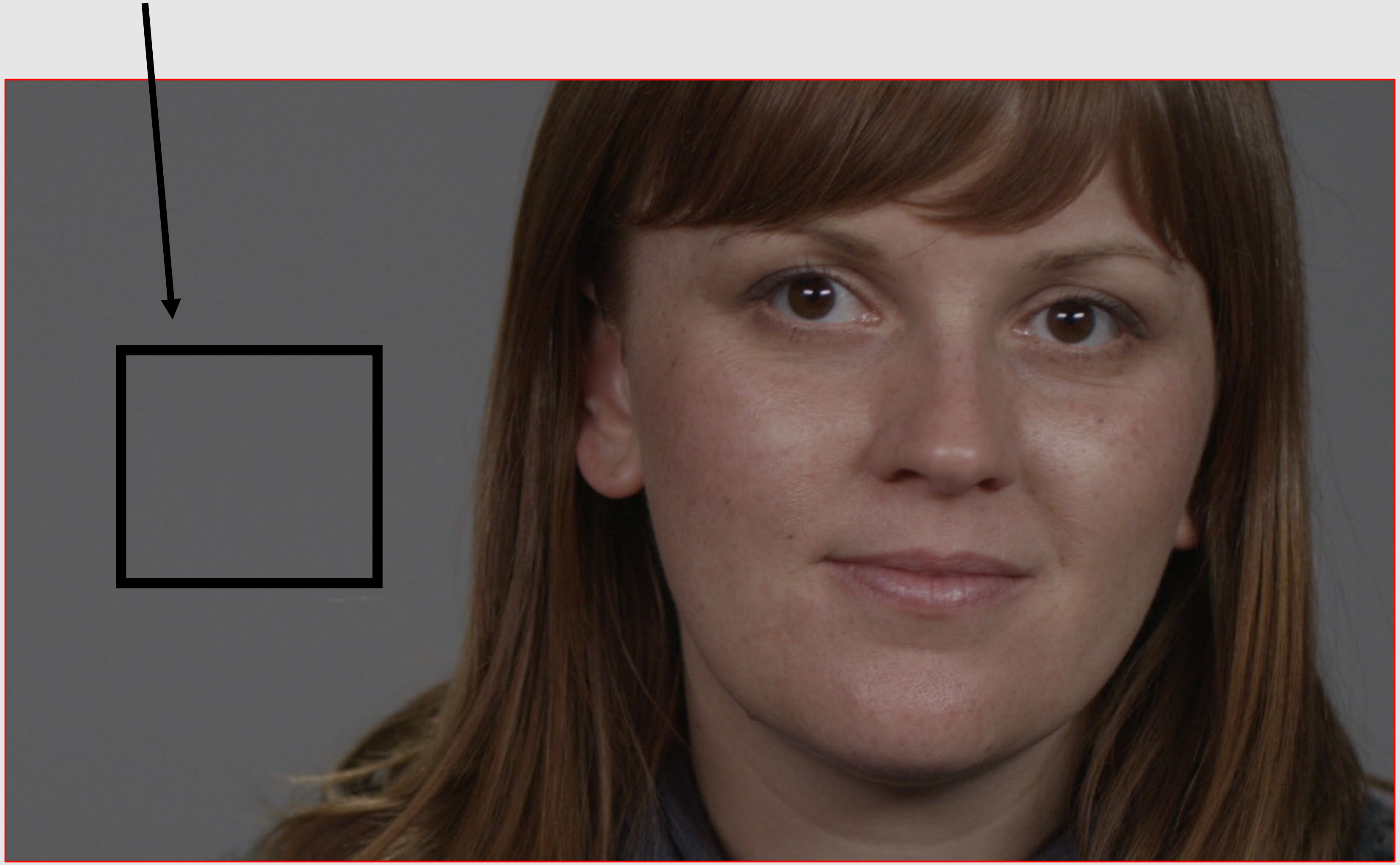
3. At every pixel site pick up the colour values of the pixel at that site $[I_y, I_u, I_v]$ and then measure the "error energy"

$$E(\mathbf{x}) = \frac{(I_y - \bar{B}_y)^2}{2\sigma_y^2} + \frac{(I_u - \bar{B}_u)^2}{2\sigma_u^2} + \frac{(I_v - \bar{B}_v)^2}{2\sigma_v^2} \qquad (1)$$

4. If $E(\cdot) < E_t$ set $\alpha = 0$ Otherwise $\alpha = 1$

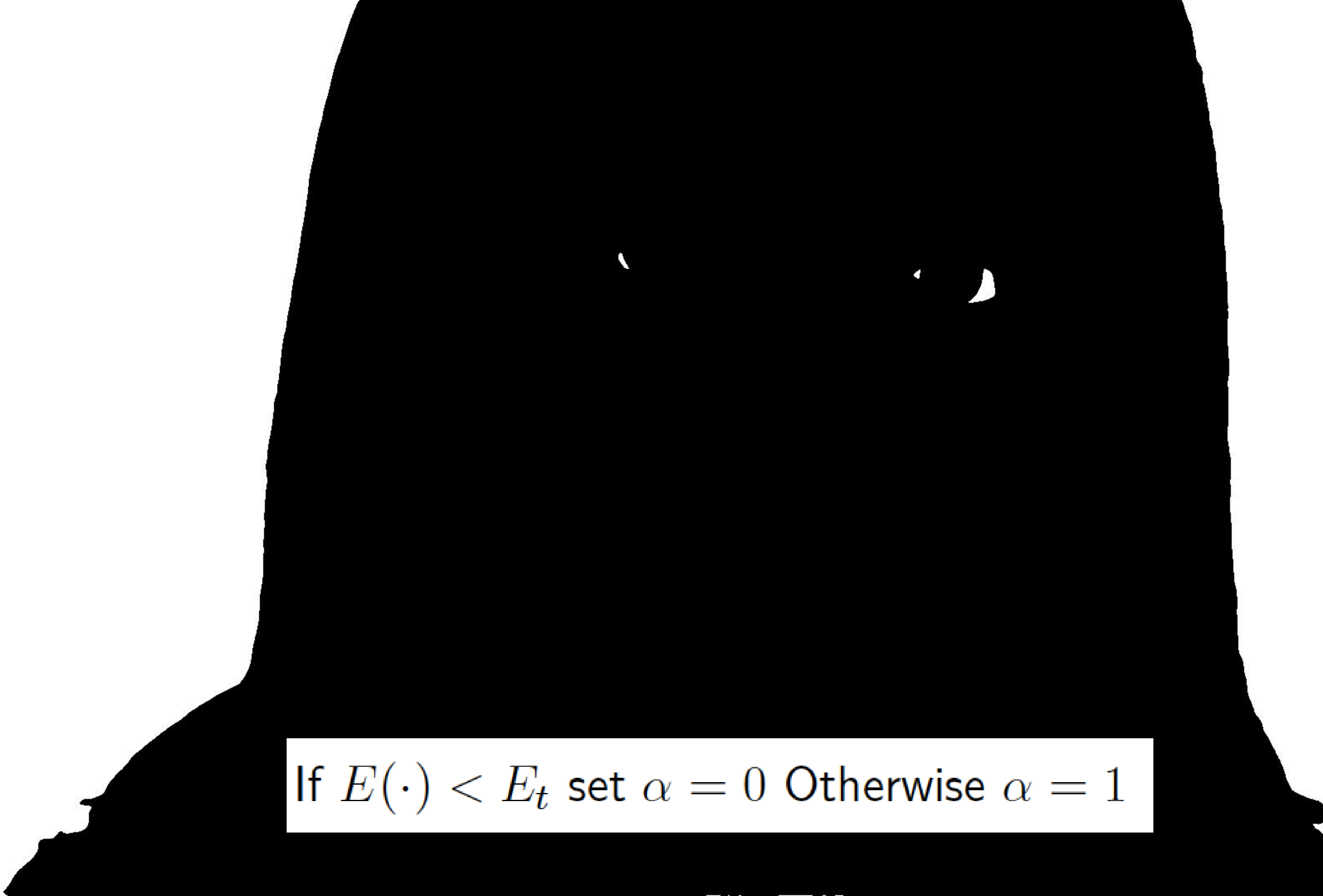
Let’s see this in action

Manually set the location of this area
And measure the mean and variance of each colour plane



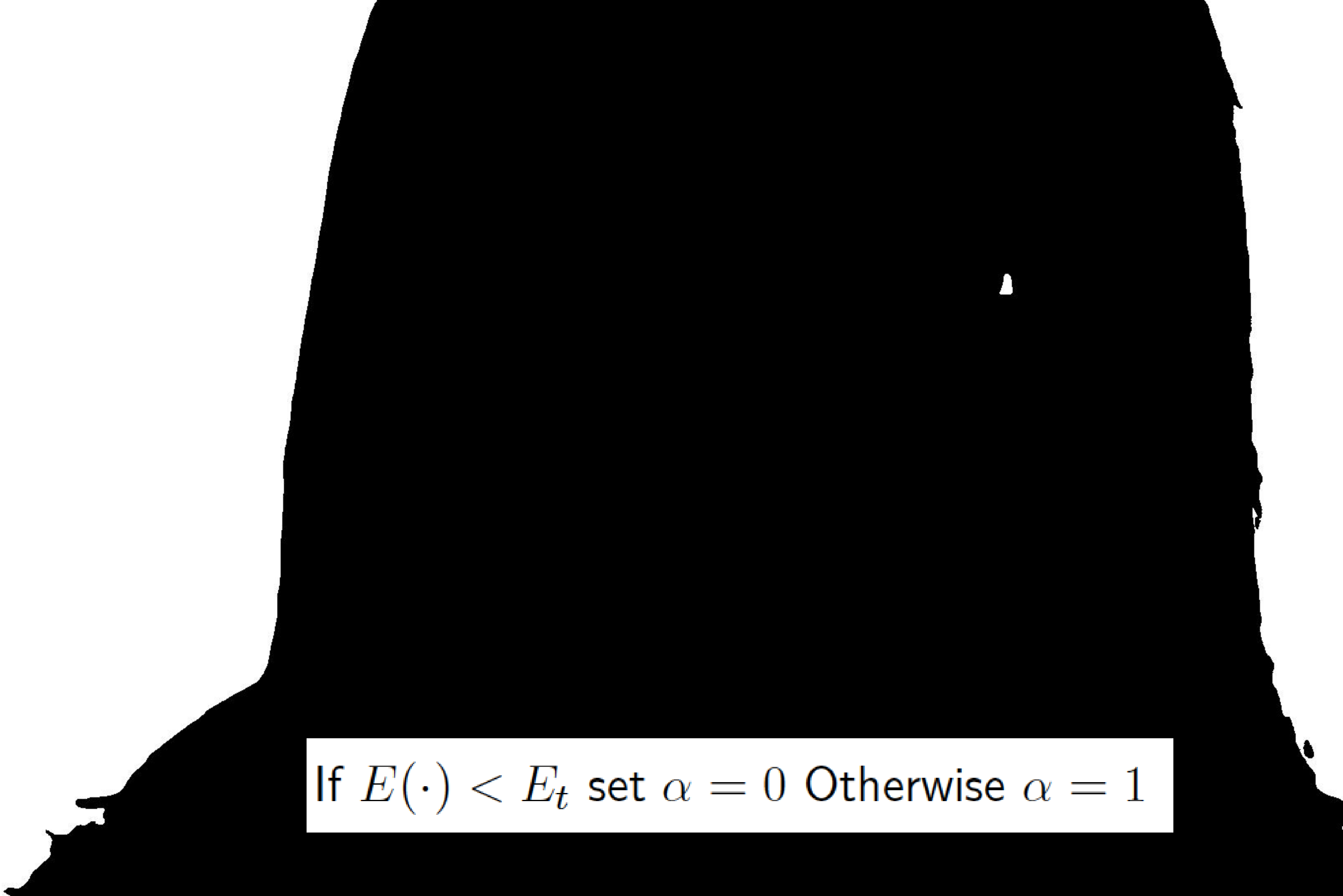
$$E(\mathbf{x}) = \frac{(I_y - \bar{B}_y)^2}{2\sigma_y^2} + \frac{(I_u - \bar{B}_u)^2}{2\sigma_u^2} + \frac{(I_v - \bar{B}_v)^2}{2\sigma_v^2}$$

$E_t = 100$



If $E(\cdot) < E_t$ set $\alpha = 0$ Otherwise $\alpha = 1$

$E_t = 40$

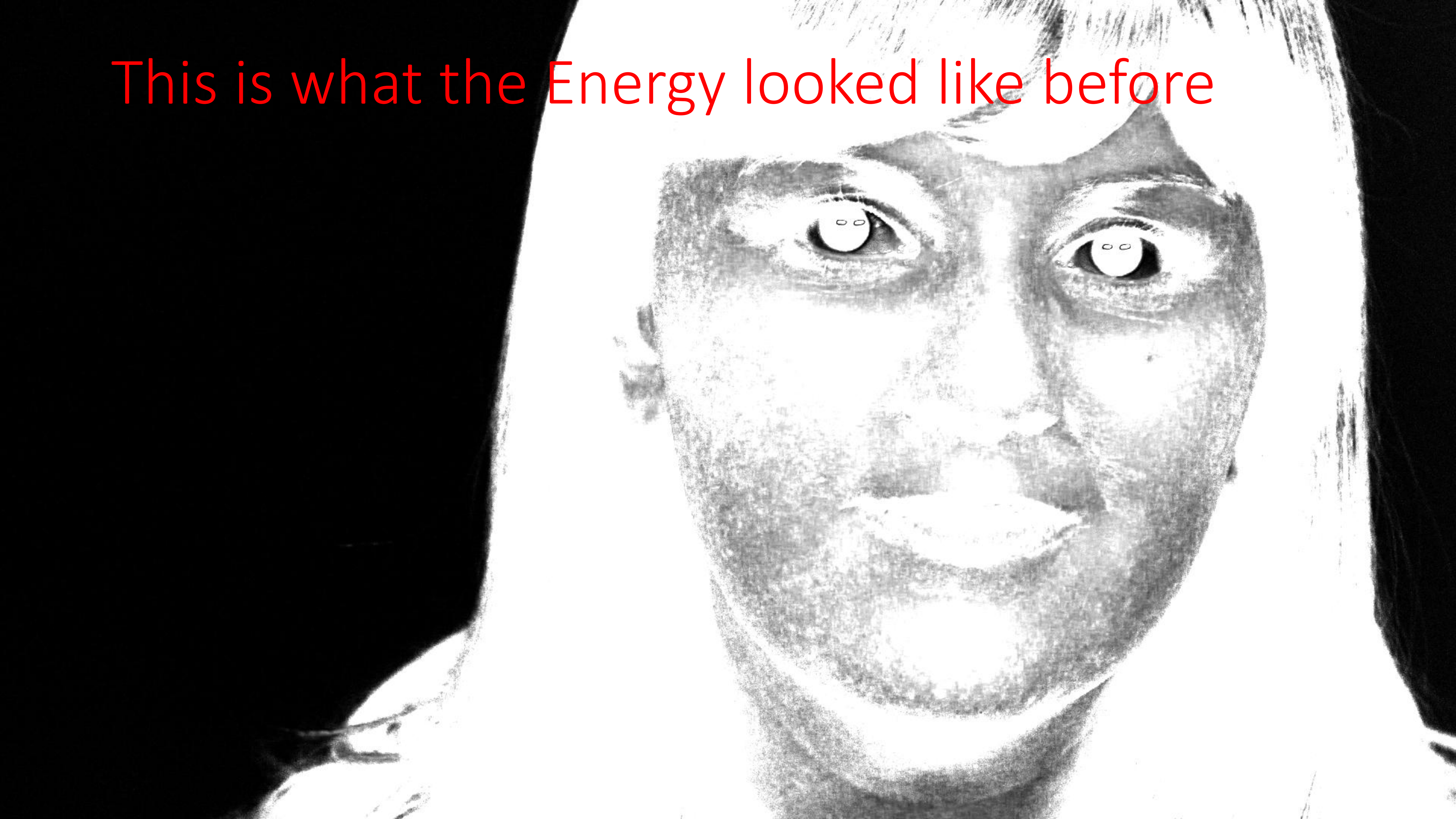


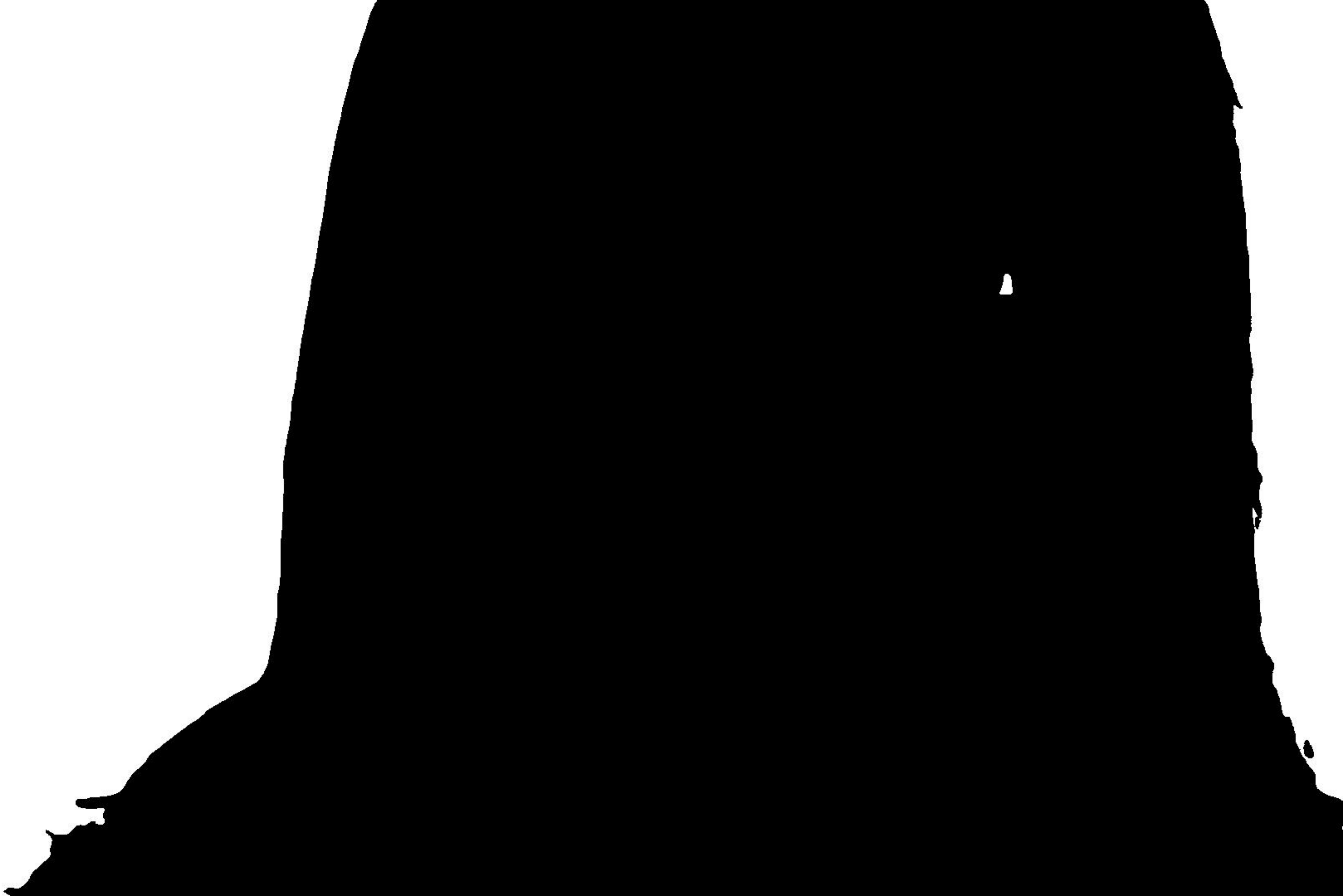
If $E(\cdot) < E_t$ set $\alpha = 0$ Otherwise $\alpha = 1$

- What’s wrong?
- How can we fix?

Smoothness is key …. Here’s a trick

Smooth the error image with an averaging filter of some kind


This is what the Energy looked like before



FIN

You will try this in a Node Graph next week

And play with filtering the energy

Many other issues : non-binary alpha important, smoothness at boundaries at the Pixel scale!