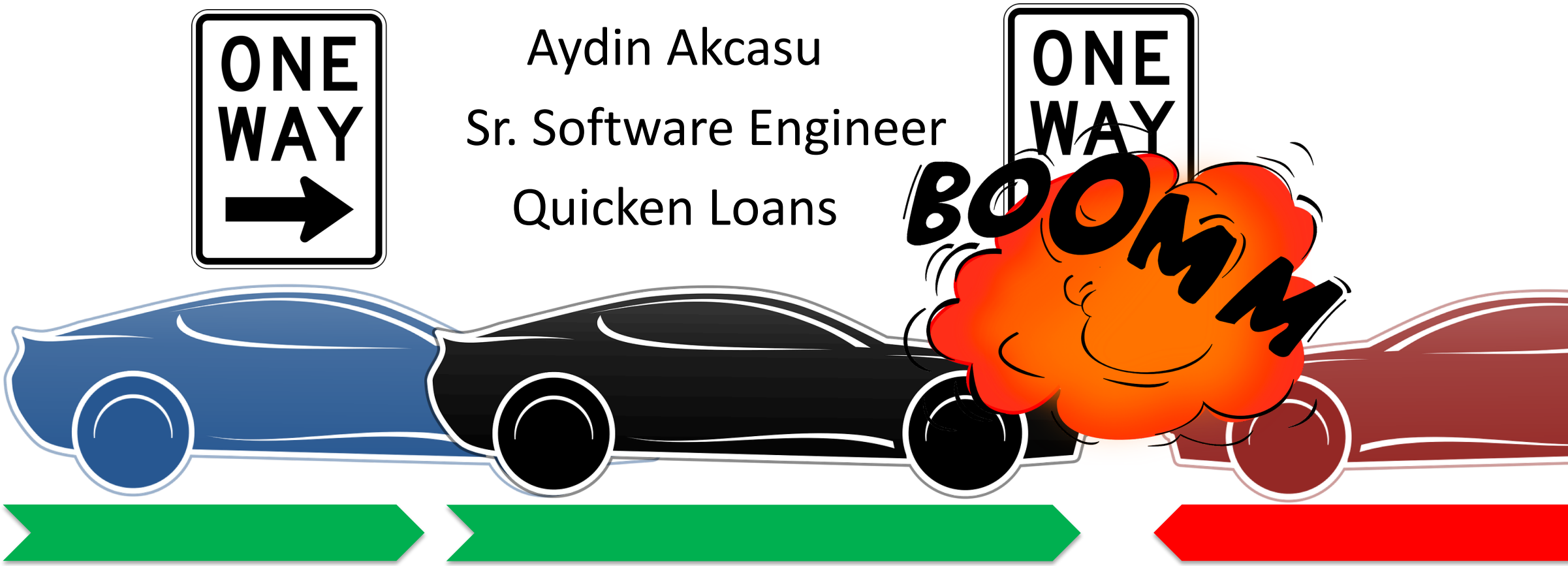
I Told You to Stand Still !
(Web Motion Detection with JavaScript)
Create A Custom Web Security Camera

Aydin Akcasu

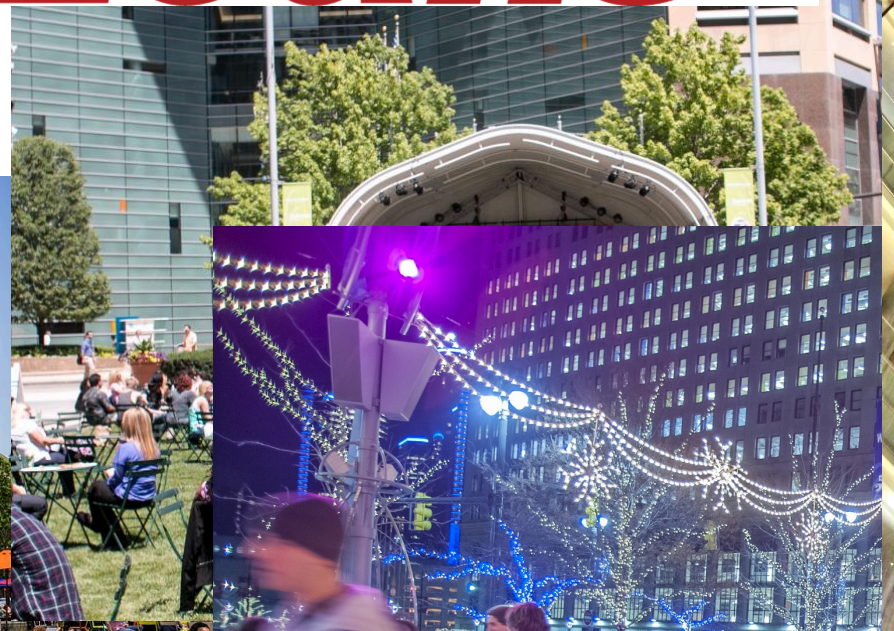
Sr. Software Engineer

Quicken Loans



- We are hiring!
- Detroit, Pheonix
- AND Remote
- Let me know!

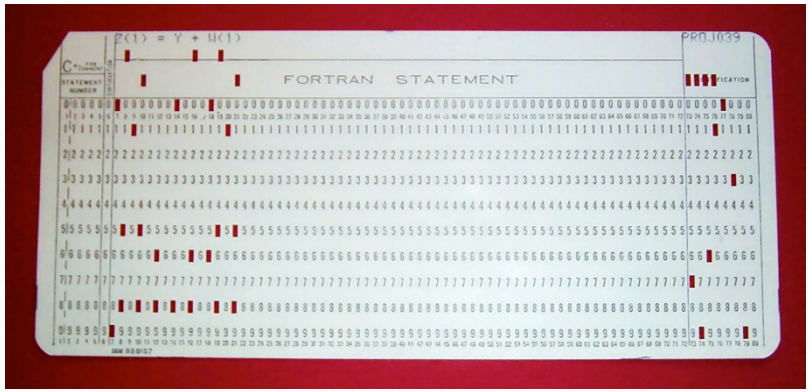
Quicken Loans®



30+ years $$$

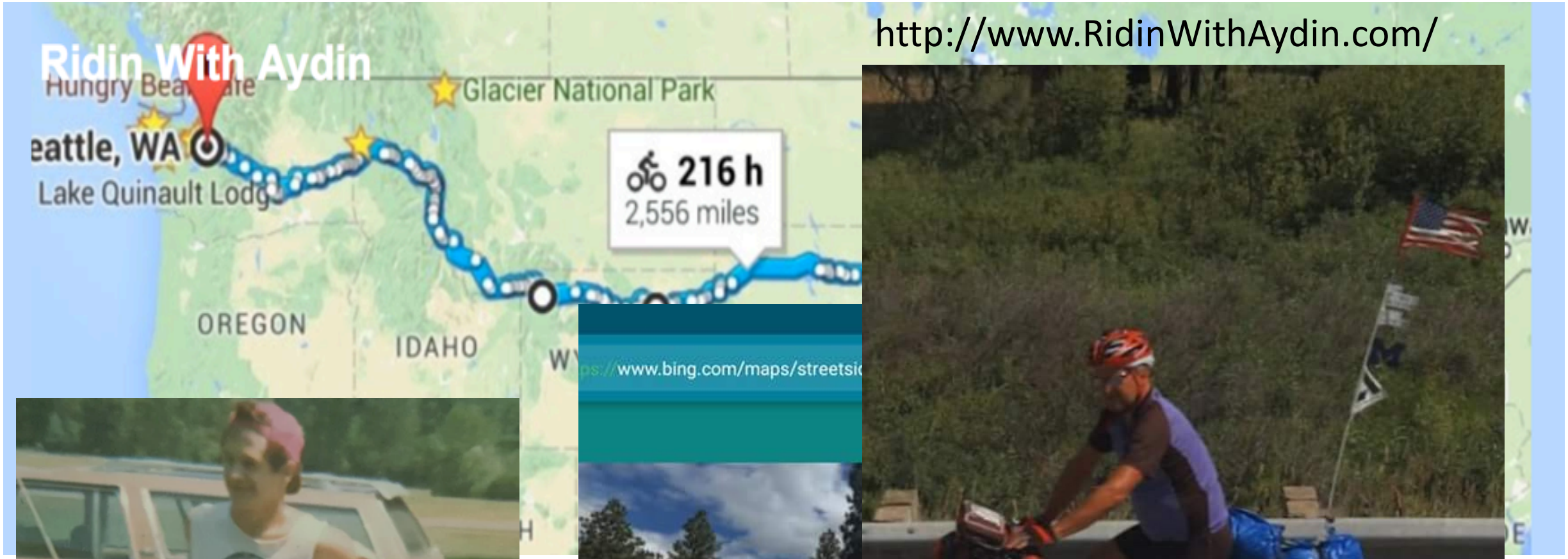
10+ years ~~$$$~~

$300, 1973==$1,700

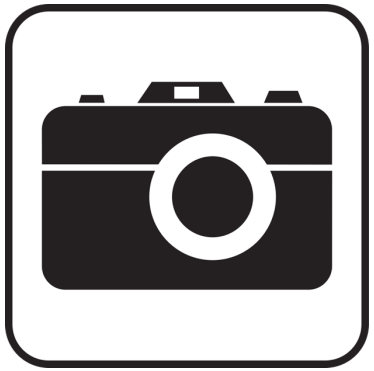


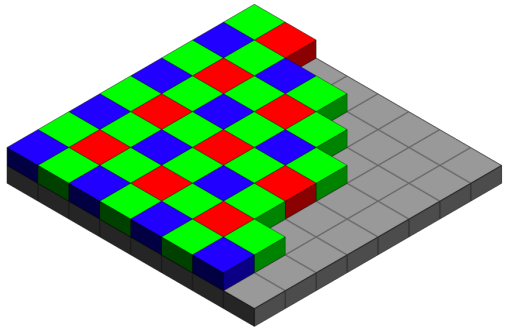
$10 K, 1983==$25K

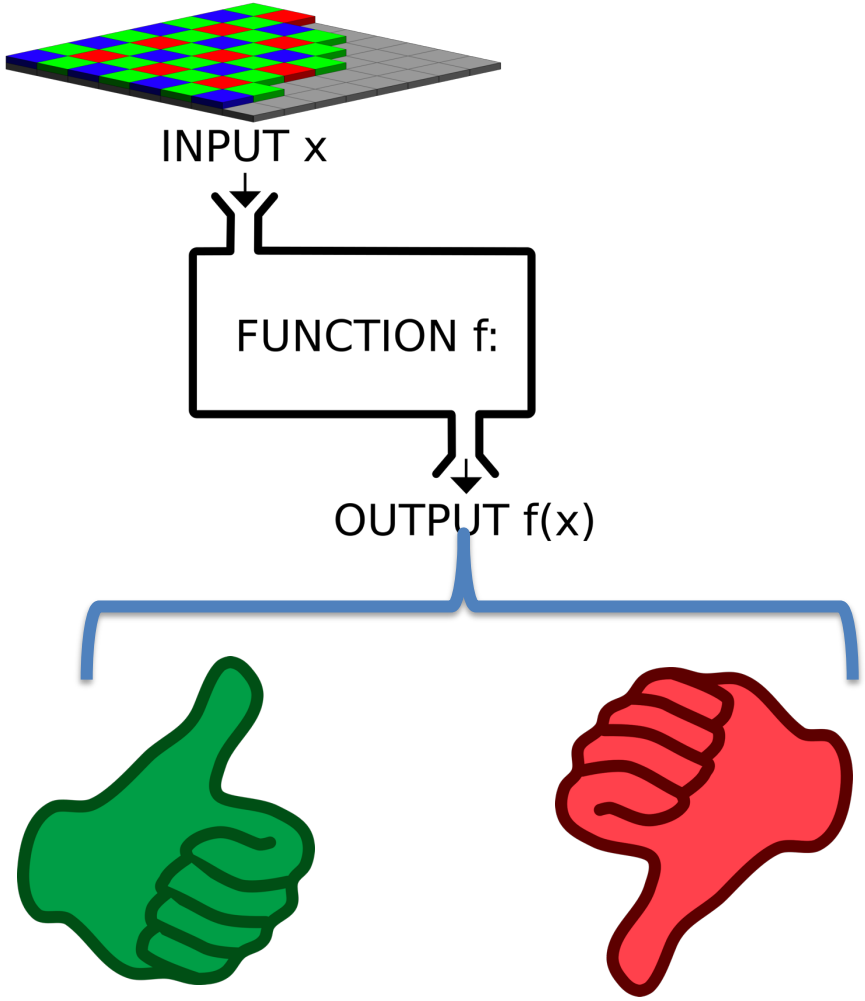
http://www.RidinWithAydin.com/



What you will learn





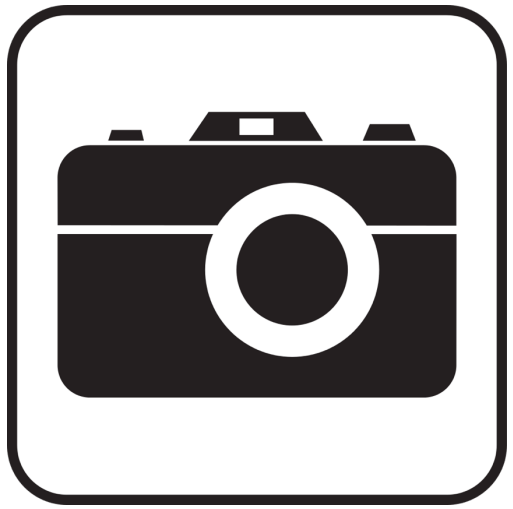


Did you know the Web* **Supports Image Processing?**

Runs on ALL* Devices

navigator.mediaDevices.*

Getting Started



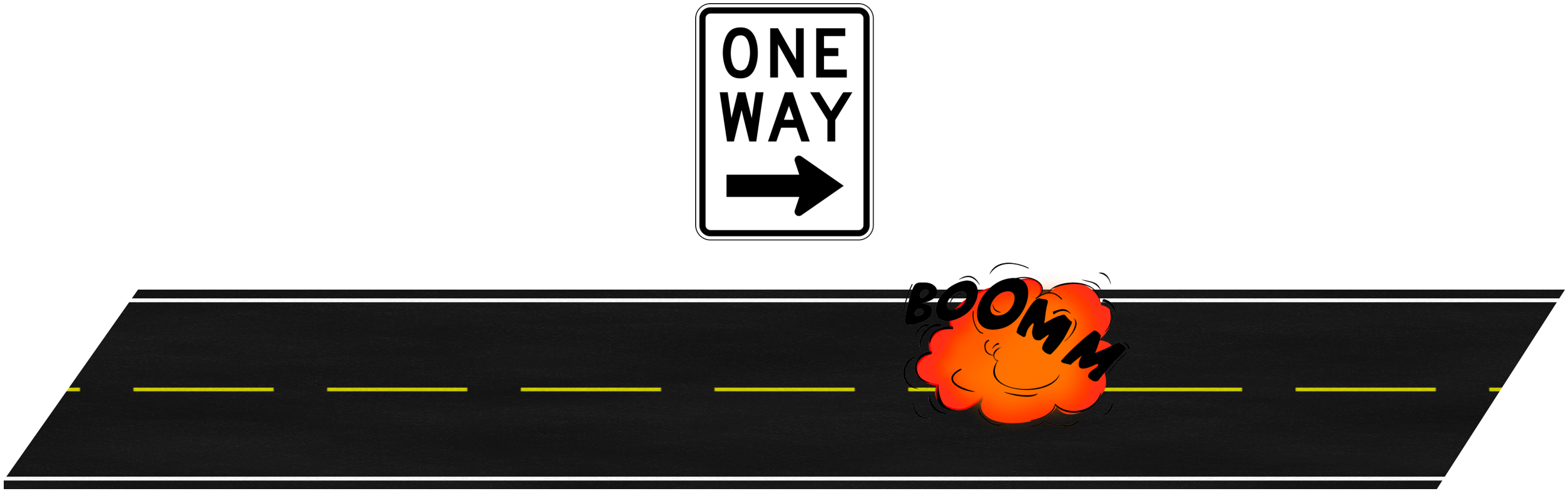
Camera (laptop or external)

A Browser (Chrome, Mozilla Firefox, Opera)

Text Editor

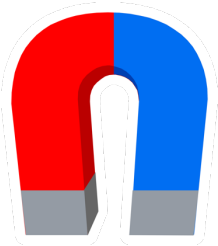
TO DEMONSTRATE THIS, HERE IS OUR PROBLEM …

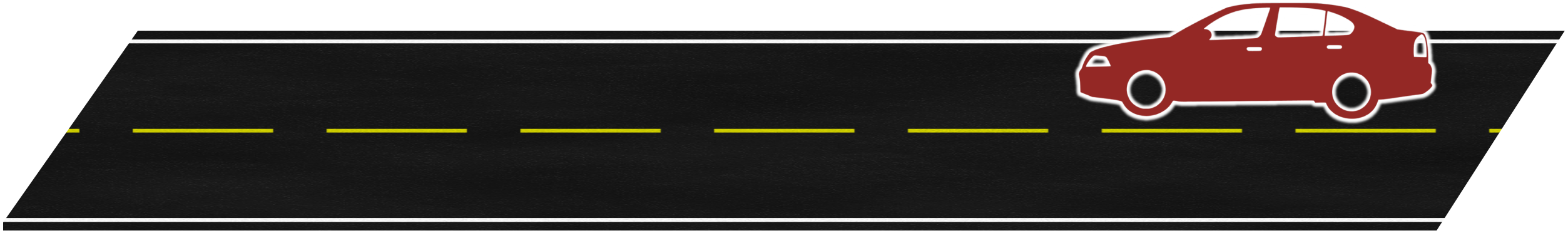
What we don’t want



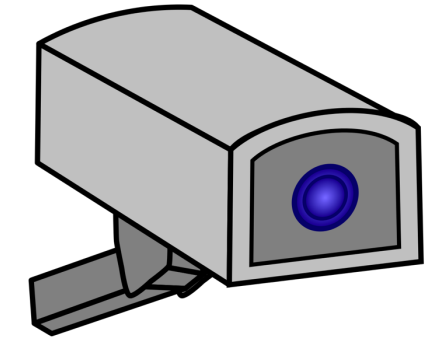
Prefer

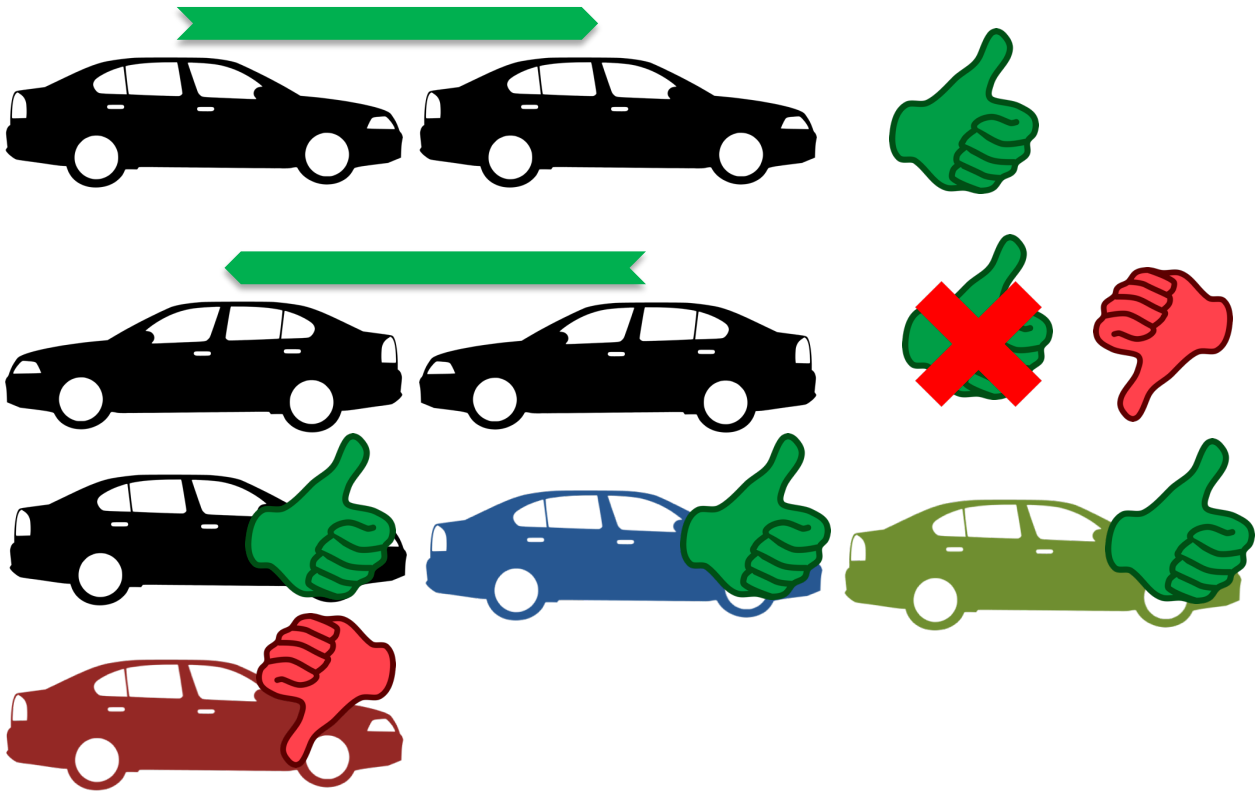
Send a tweet
Send a text
Send a photo of the car to the police
Email
Set off an alarm
Bell

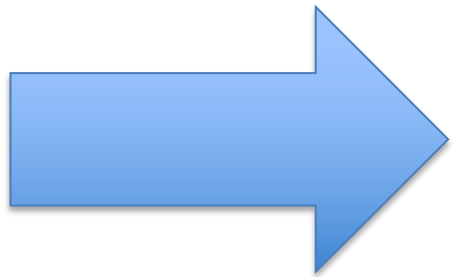




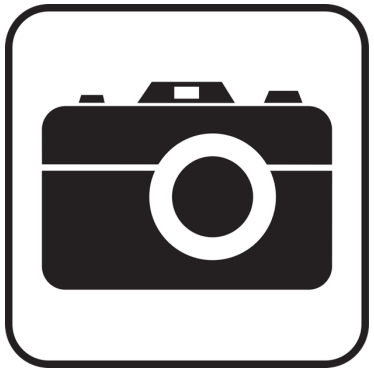
Options today

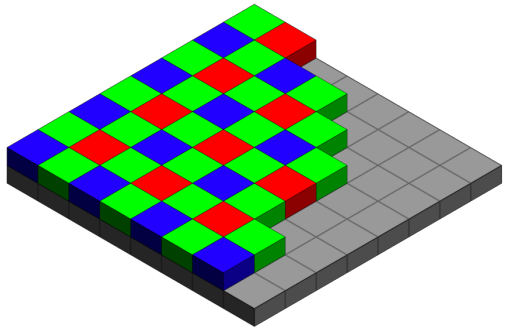


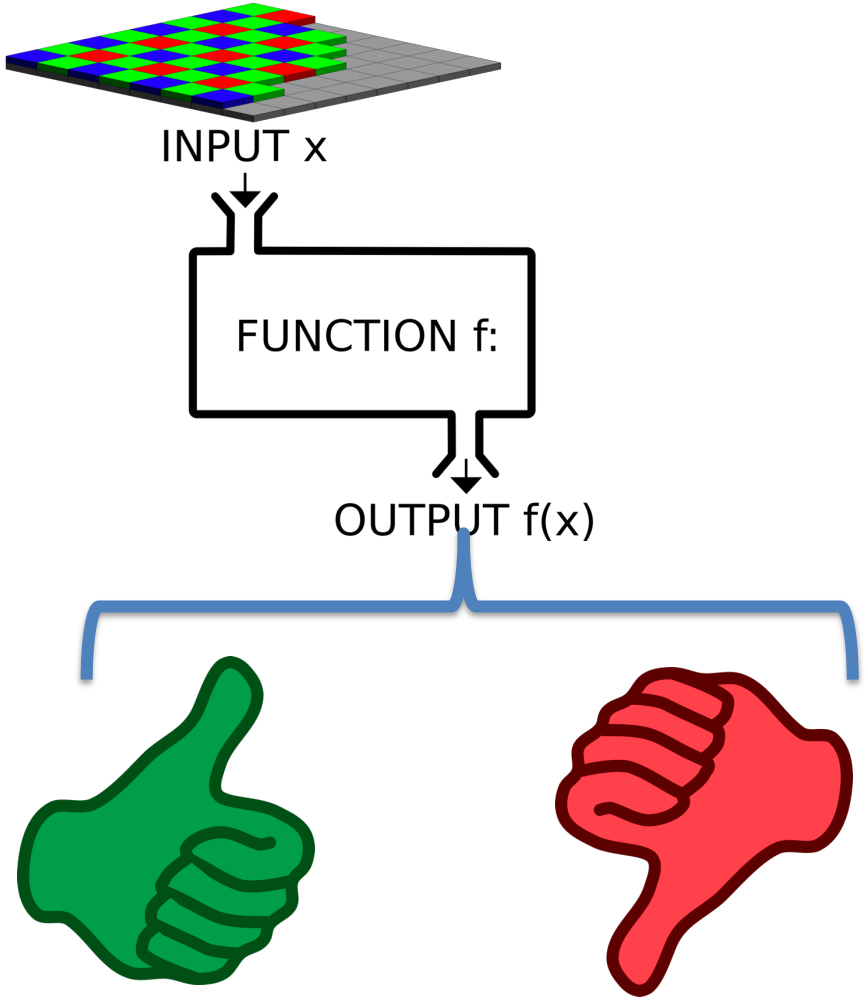




What you will learn







Terminology

- Image
- Resolution
- Image Types
- Pixels
- Colors
- Accessing image data

WHAT IS AN IMAGE ANYWAY?

Intro to Bytes…

- Byte
  - 8 Bits (0,1), or 2^8== 256 possibilities
  - 0-255 (base 10), or Decimal
  - 0x00-0xFF (base 16), or Hex
- RGB
  - 3 Bytes: Red, Green, Blue

Decimal	Hex Digits
0	0x0
1	0x1
2	0x2
.	0x…
.	0x…
.	0x…
9	0x9
10	0xA
11	0xB
12	0xC
13	0xD
14	0xE
15	0xF
16	0x10

What is an Image?

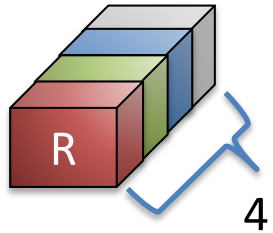
5x5 Image

(5 Pixels x 5 Pixels)

(Resolution is 5x5)

(Order of the Pixels)

0	1	2	3	4
5	6	7	8	9
10	11	12	13	14
15	16	17	18	19
20	21	22	23	24



4 Bytes = Red, Green, Blue,
Alpha (Transparency)

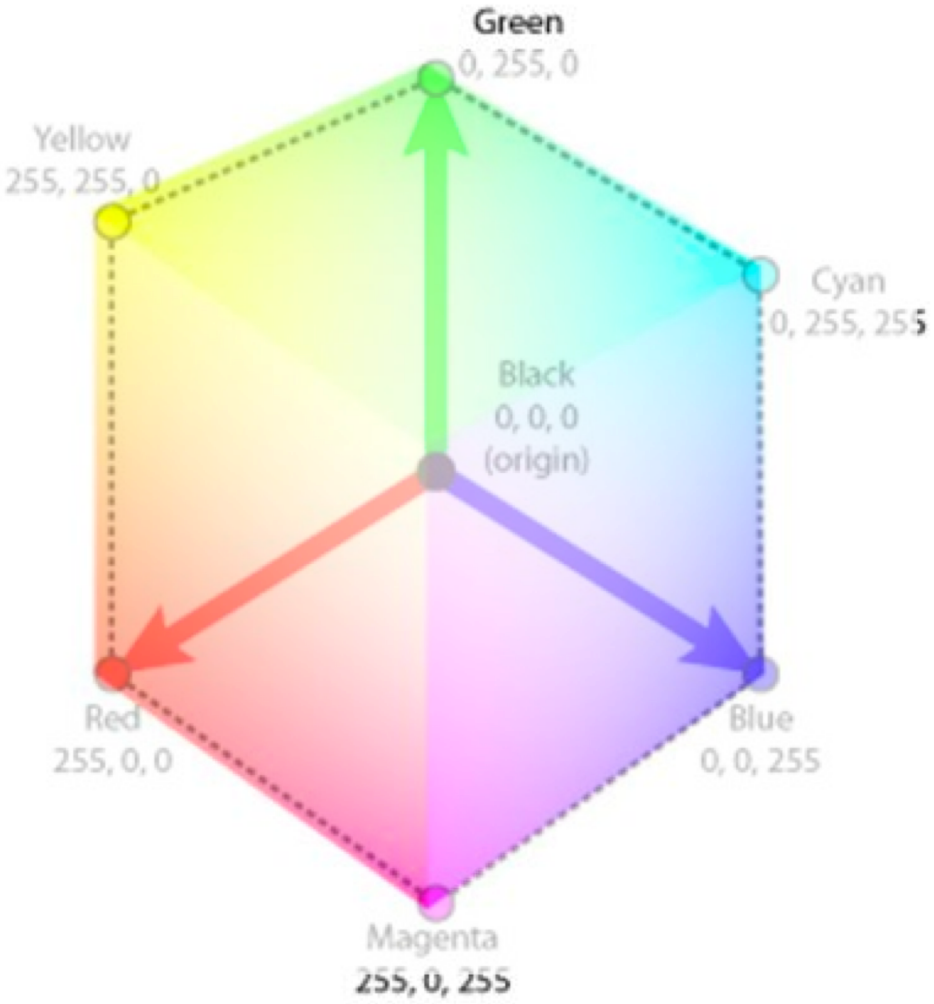
0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24

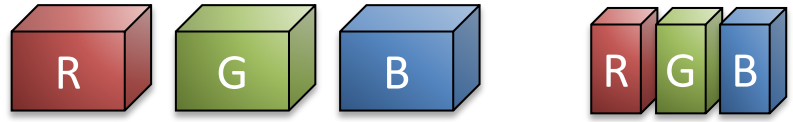
Offset = (5* row + column)

Byte Offset = (5* row + column) * 4

TYPES OF IMAGES

Color: RGB Details

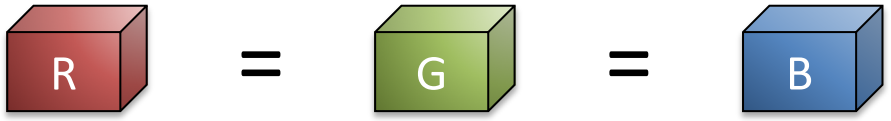


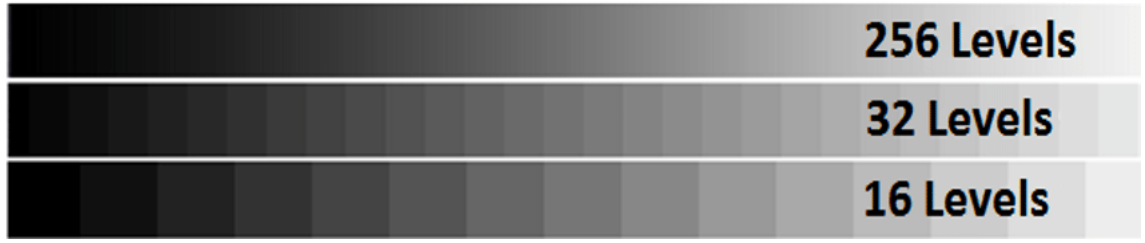


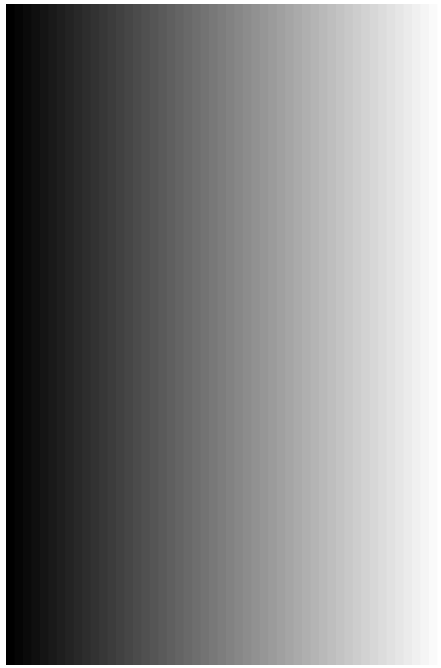
RR GG BB RRGGBB

```
Red:     (0xFF,0x00,0x00) = 0xFF0000
Green:   (0x00,0xFF,0x00) = 0x00FF00
Blue:    (0x00,0x00,0xFF) = 0x0000FF
White:   (0xFF,0xFF,0xFF) = 0xFFFFFF
Black:   (0x00,0x00,0x00) = 0x000000
Magenta: (0xFF,0x00,0xFF) = 0xFF00FF
Cyan:    (0x00,0xFF,0xFF) = 0x00FFFF
Yellow:  (0xFF,0xFF,0x00) = 0xFFFF00
```

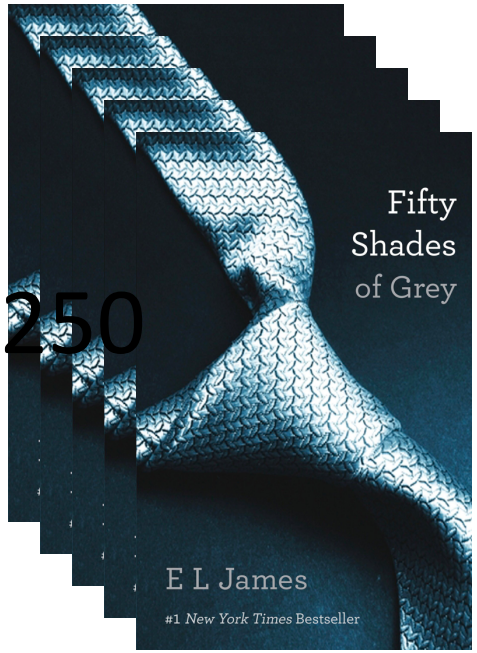
Greyscale, Greyscale, or Black and White (BW):

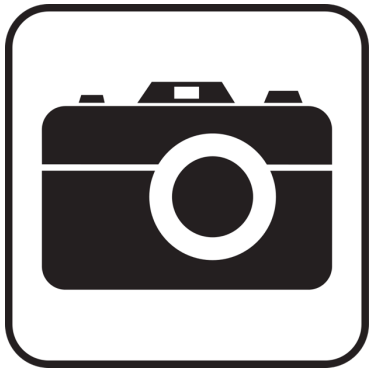


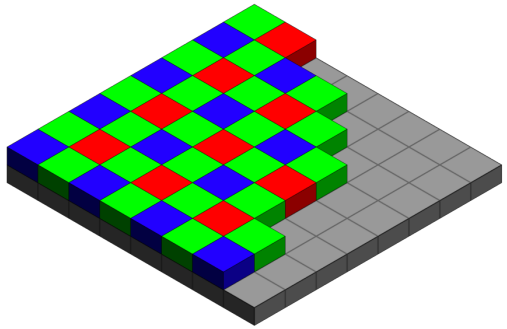




> 250







OVERVIEW OF CONNECTING TO A CAMERA

Basic Capture

```
<video id="video" autoplay></video>

<script>
    async function initializeVideo(videoElement) {
        var params = {
            audio: false,
            video: { width: 320, height: 240 }
        };

        videoElement.srcObject = await
        navigator.mediaDevices.getUserMedia(params);
        // Next Step
    }
    var video = document.getElementById('video');
    initializeVideo(video);
</script>
```

Constaints

For security

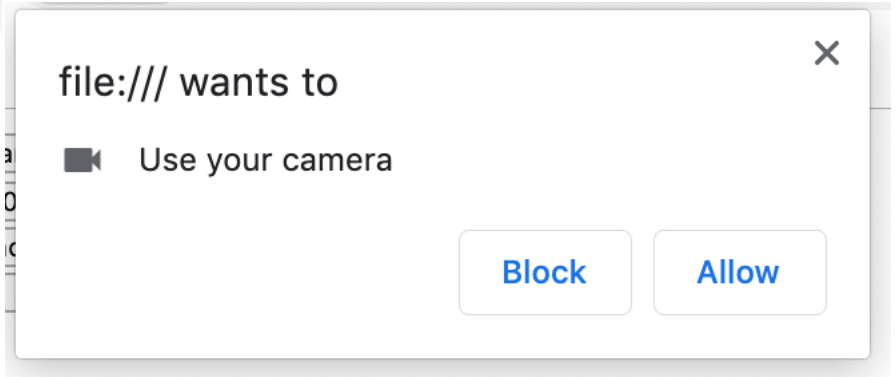


Image Capture

```
<video id="video" autoplay></video>

<script>
    async function initializeVideo(videoElement) {
        var params = {
            audio: false,
            video: { width: 320, height: 240 }
        };

        videoElement.srcObject = await
        navigator.mediaDevices.getUserMedia(params);
        initializeImageCapture();
    }
    var video = document.getElementById('video');
    initializeVideo(video);
</script>
```

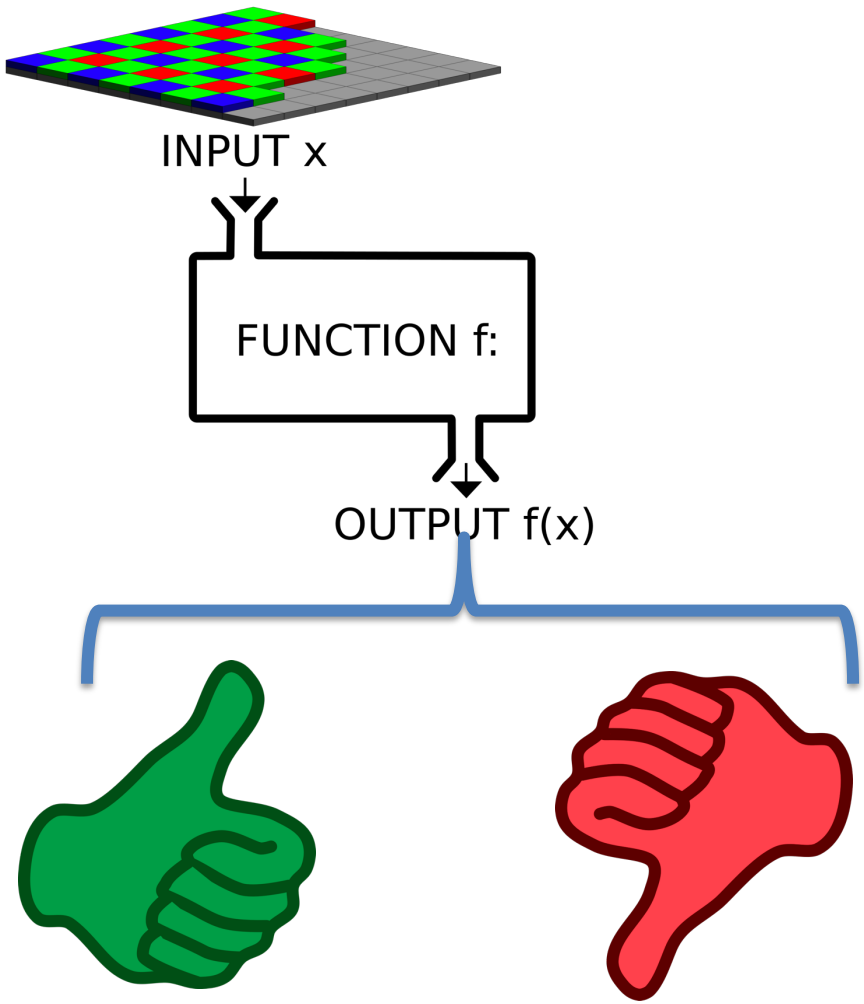
Magic happens here!

initializeImageCapture()

```
function initializeImageCapture() {
    var canvas = document.getElementById('canvas');
    canvas.width = 480; canvas.height = 240;

    var context = canvas.getContext('2d');
    setInterval(capture, 100);

    function capture() {
        context.drawImage(video, 0, 0, 320, 240);
        const getImage = context.getImageData(0, 0, 320, 240);

        const data = getImage.data;
        // Data Processing
        context.putImageData(getImage, 0, 0);
    }
}
```

Magic happens here!

IMAGE PROCESSING



Data Processing (Dim Image)

```
for (let y = 0; y < 240; y++) {
    for (let x = 0; x < 320; x++) {
        var i = 4 * (y * 320 + x);

        data[i + 0] = data[i + 0] / 2;
        data[i + 1] = data[i + 1] / 2;
        data[i + 2] = data[i + 2] / 2;
    }
}
```

Byte Offset

RGB Offsets

Dim Image

Data Processing (Gray Image)

```
// Data Processing
for (let y = 0; y < 240; y++) {
    for (let x = 0; x < 320; x++) {
        var i = 4 * (y * 320 + x);

        const gray =
                ( data[i + 0]
                + data[i + 1]
                + data[i + 2]
                ) / 3;
        data[i + 0] = gray;
        data[i + 1] = gray;
        data[i + 2] = gray;
    }
}
```

Average RGB

Data Processing (Negative Image)

```
// Data Processing
for (let y = 0; y < 240; y++) {
    for (let x = 0; x < 320; x++) {
        var i = 4 * (y * 320 + x);

        data[i + 0] = 0xFF - data[i + 0];
        data[i + 1] = 0xFF - data[i + 1];
        data[i + 2] = 0xFF - data[i + 2];
    }
}
```

Inverts Image

(255-x)

If x=0, result is 255

If x=255, result is 0

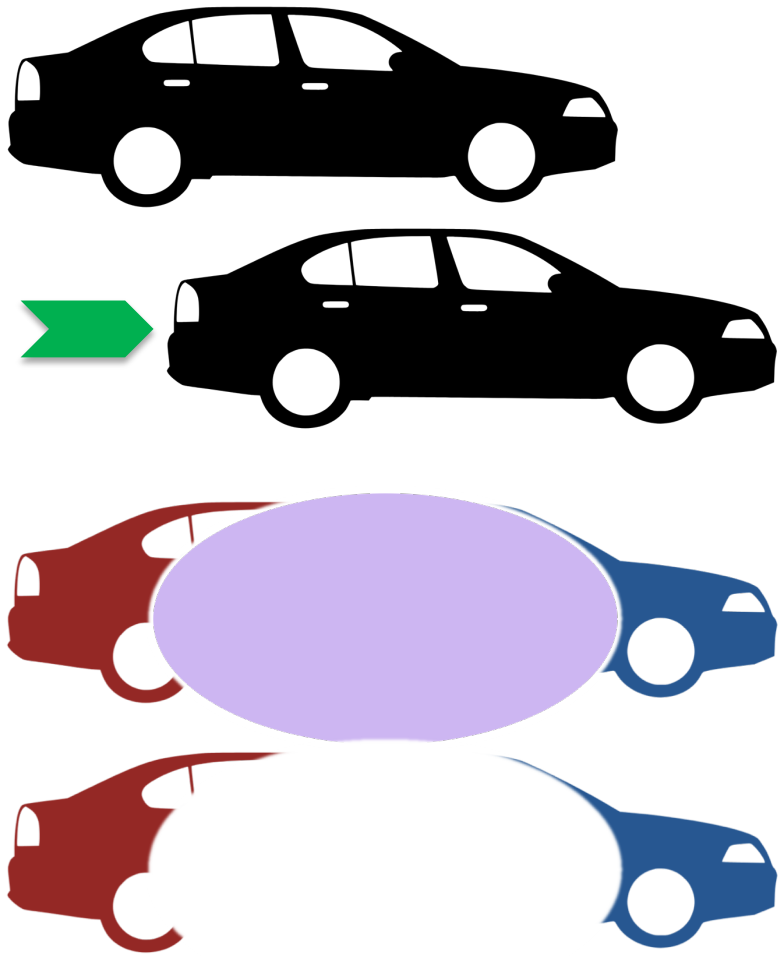
If x= 25, result is 230

Two Image Capture

```
function capture() {
    context.drawImage(video, 0, 0, 320, 240);

    const tempImage    = context.getImageData(0, 0, 320, 240);
    const currentImage = context.getImageData(0, 0, 320, 240);

    const currentData  = currentImage.data;
    const previousData = (previousImage || currentImage).data;

    // Process: imageDifferences(currentData, previousData);

    context.putImageData(currentImage, 0, 0);
    previousImage = tempImage;
}
```

Magic happens here!

How can you show motion?



imageDifferences

```
function imageDifferences(data, previousData) {
    const toGray = (d, i) => (d[i + 0] + d[i + 1] + d[i + 2]) / 3;

    for (let y = 0; y < 240; y++) {
        for (let x = 0; x < 320; x++) {
            var i = 4 * (y * 320 + x);

            const currentPixel  = toGray(data, i);
            const previousPixel = toGray(previousData, i);

            data[i + 0] = 2 * (previousPixel - currentPixel);
            data[i + 1] = 0x00;
            data[i + 2] = 2 * (currentPixel - previousPixel);
        }
    }
}
```

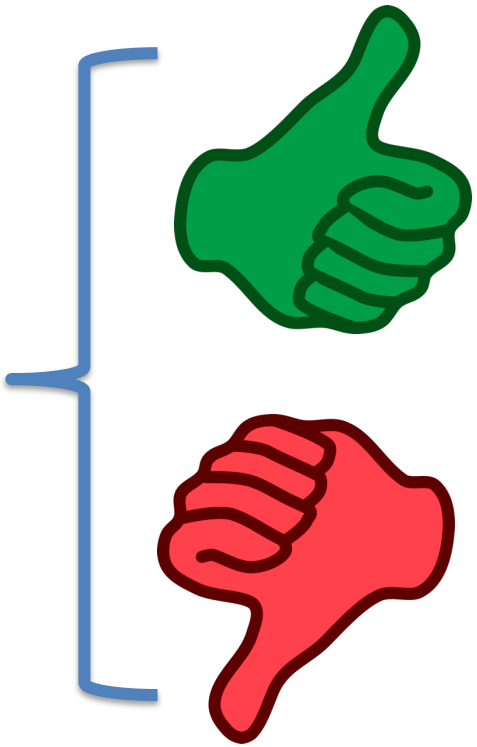
Red: In ‘from’ area

Green: 0x00, black

Blue: In ‘to’ area

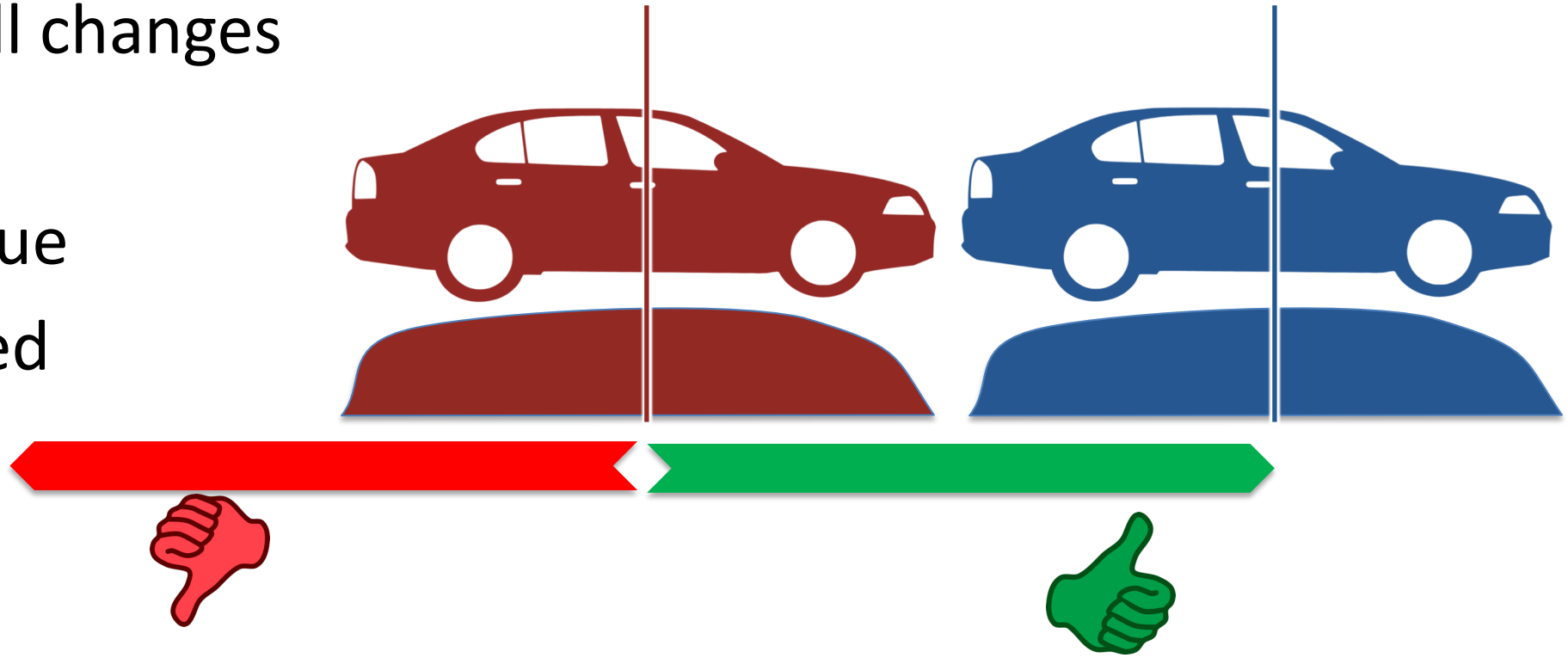
If no movement: all black

DECISIONS, DECISIONS, DECISIONS,

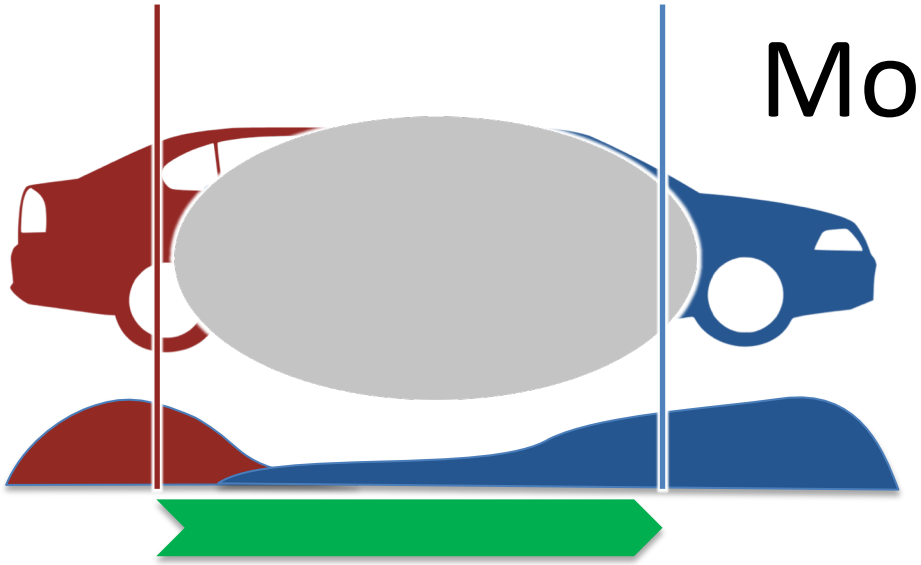


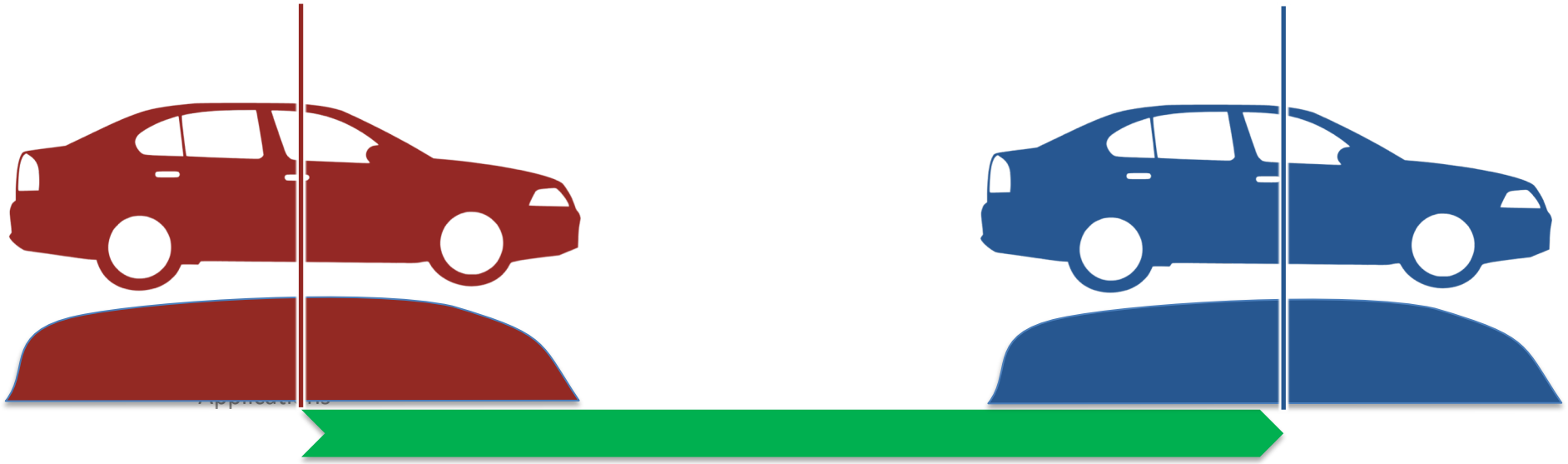
So how can we determine if something has moved

- Count of all changes
- Histogram
- Average Blue
- Average Red
- Difference



More Examples





THE DEMO

Links:

- Live Ascii Art:
  - idevelop.github.com/ascii-camera
- Other Links:
  - https://andrei.codes/predator-vision/
  - https://webrtchacks.com/
  - https://jk-lee.com/generative-design-book/

Aydin Akcasu

Email: AydinAkcasu@QuickenLoans.com

AAkcasu@gmail.com

LinkedIn: Linkedin.com/in/aydin-akcasu-51063

Twitter: @AAkcasu

Trip Info: RidinWithAydin.com

THE END

Hacking Bluetooth Devices and Controlling Them with Your Browser

Aydin Akcasu

- AAkcasu@gmail.com
- **linkedin.com/in/aydin-akcasu-51063**
- **@**AAkcasu

All slides, everything

- https://aydinakcasu.github.io/WebComputerVision/
- We are hiring, Detroit + Remote:
- https://quickenloanscareers.com/
- Send me your resume…
  - AydinAkcasu@QuickenLoans.com

TODO:

- Put on Github
- Put add. Github site…
- Test on phone
- https://andrei.codes/predator-vision/

- “I told you not to move”, add sound
- Send beep if moved
- (DONE) Add borderless display
- (DONE) Intro slide Web Page