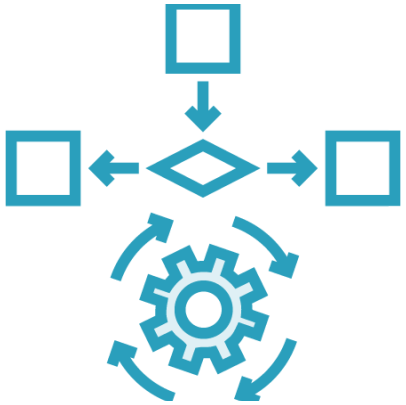
Utilizing the SIFT and HOG Algorithms for Feature Detection

David Tucker
TECHNICAL ARCHITECT & CLOUD CONSULTANT
@_davidtucker_ davidtucker.net

Types of Computer Vision Approaches



Non-learning

Configurable algorithms

Learning

Neural networks

Overview

Introducing image processing techniques

Exploring feature descriptors with the SIFT algorithm

Utilizing SIFT feature descriptors for matching

Exploring feature descriptors with the HOG algorithm

Utilizing HOG feature descriptors for detecting the presence of an object

Image Processing Techniques

Kernel

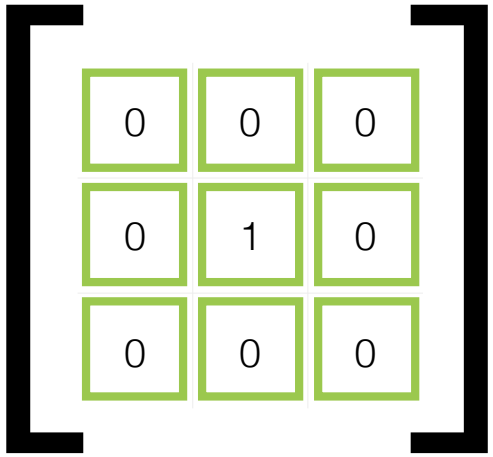
A matrix that is applied to an image channel's values for the purpose of applying a filter such as a Gaussian blur or extracting a specific feature such as an object's edges.

Convolution

The process by which a new value is calculated for a pixel based on adding the original pixel value and all surrounding pixels after a kernel has been applied.


YAMAHA
Epiphone

Identity Kernel



219 X 0 = **0**	184 X 0 = **0**	148 X 0 = **0**
242 X 0 = **0**	217 X 1 = **217**	187 X 0 = **0**
248 X 0 = **0**	240 X 0 = **0**	222 X 0 = **0**

Original Pixel Value = **217**

New Pixel Value = **217**


YAMAHA
Epiphone

Gaussian Blur 3x3 Kernel

$$\frac{1}{16}\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

219 X 1 = **219**	184 X 2 = **368**	148 X 1 = **148**
242 X 2 = **484**	217 X 4 = **434**	187 X 2 = **374**
248 X 1 = **248**	240 X 2 = **480**	222 X 1 = **222**

Original Pixel Value = **217**

New Pixel Value = **186**

Gaussian Blur 5x5 Kernel

$$\frac{1}{256}\begin{bmatrix} 1 & 4 & 6 & 4 & 1 \\ 4 & 16 & 24 & 16 & 4 \\ 6 & 24 & 36 & 24 & 6 \\ 4 & 16 & 24 & 16 & 4 \\ 1 & 4 & 6 & 4 & 1 \end{bmatrix}$$

Edge Detection Kernel

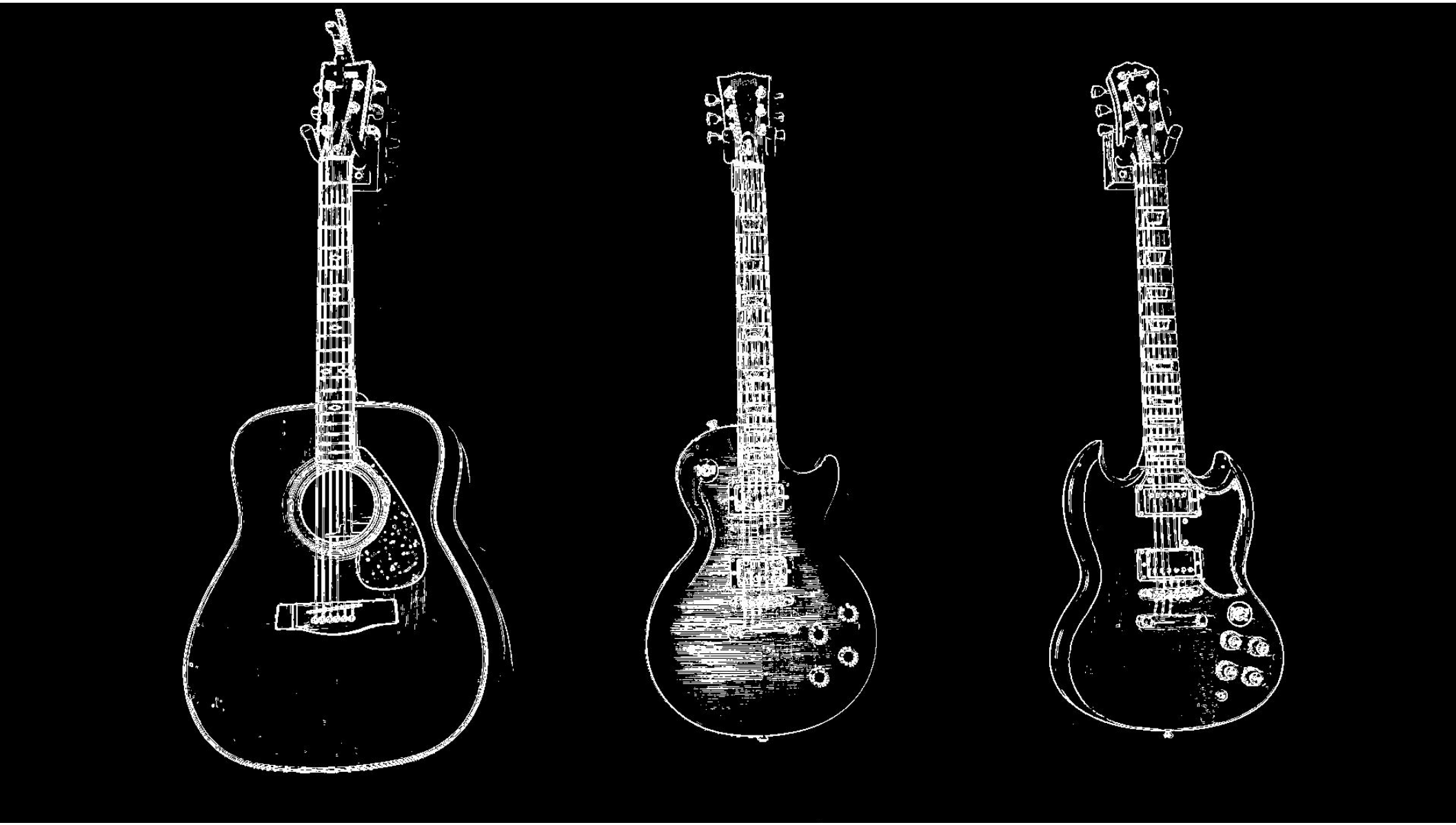
$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

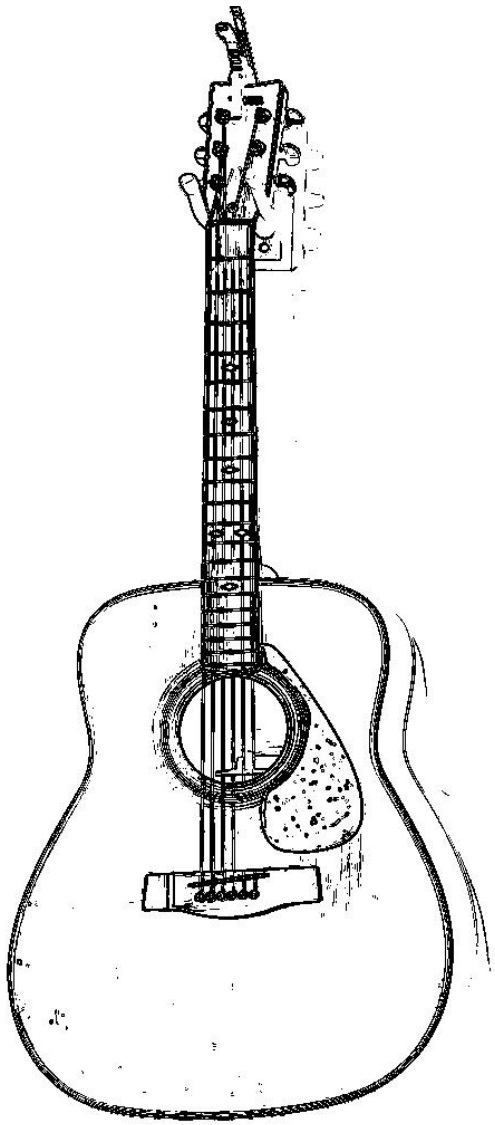
Edge Kernel X

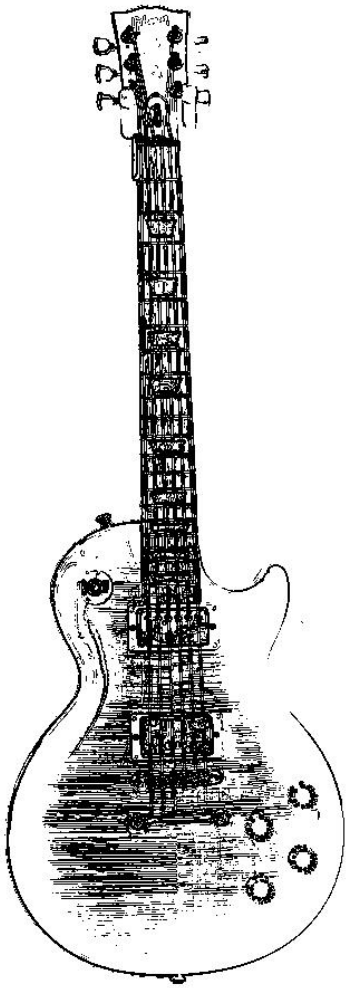
$$\begin{bmatrix} 1 & 2 & 1 \\ 2 & 4 & 2 \\ 1 & 2 & 1 \end{bmatrix}$$

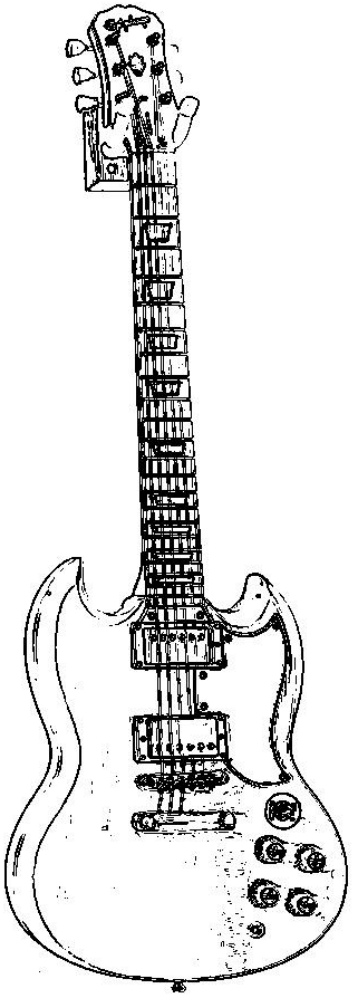
Edge Kernel Y

Convolution Value = $\sqrt{x^2+y^2}$









SIFT Introduction

“The **scale-invariant feature transform** (SIFT) is a feature detection algorithm in computer vision to detect and describe local features in images. It was patented in Canada by the University of British Columbia and published by David Lowe in 1999.”

Wikipedia

Scale-invariant

SIFT utilizes an algorithmic approach that allows for feature descriptors to correctly identify images irrespective of scale.

SIFT Algorithmic Steps

Scale-space Extrema Detection	Keypoint Localization
Orientation Assignment	Keypoint Descriptor

Scale-space Extrema Detection

Utilizes difference of Gaussian functions

Identifies keypoints that occur at multiple scales

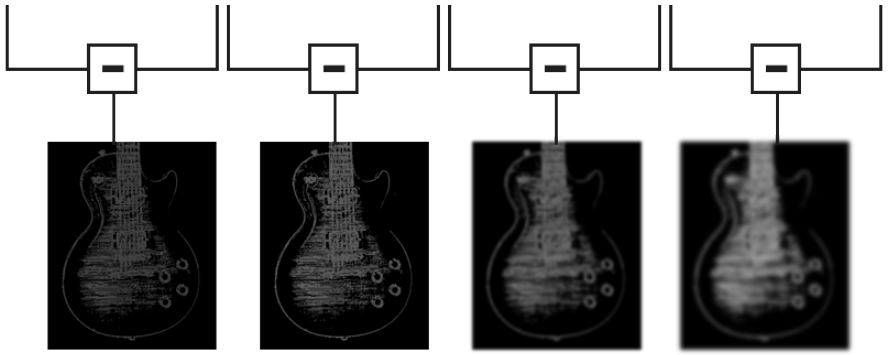
Enables potential keypoints to be scale-invariant

Remaining keypoints will be filtered in future steps

Difference of Gaussian

SIFT utilizes **difference of Gaussian** functions to identify potential keypoints in a scale invariant manner. This approach finds maxima and minima that occur at multiple scales with different sigma values for the gaussian blur.

Difference of Gaussian



Keypoint Localization

Identifies local keypoints by finding extrema based on surrounding pixels

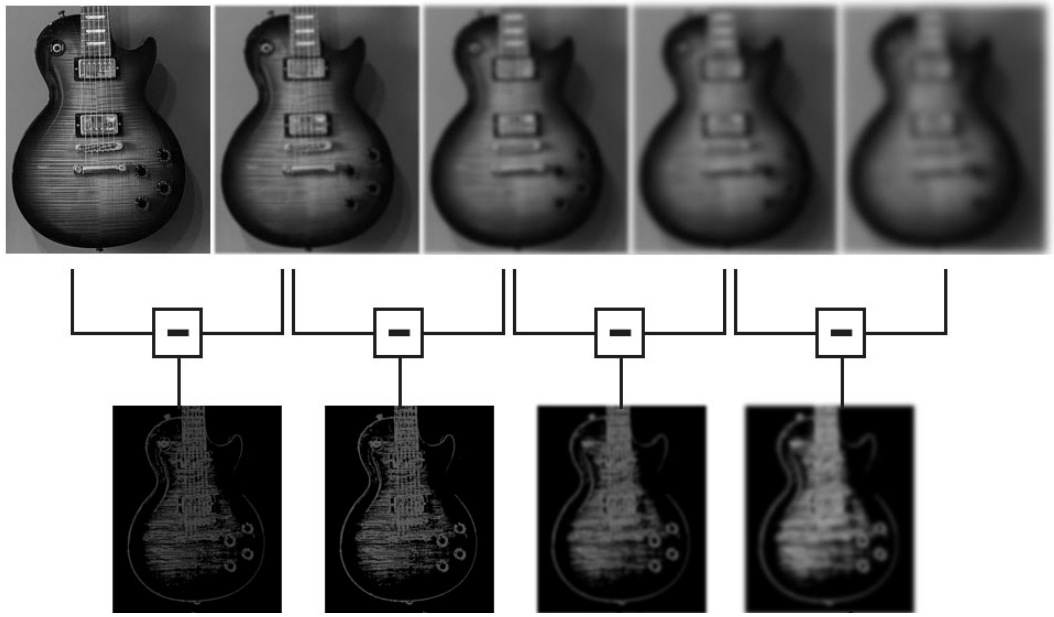
Compares the keypoint with surrounding values in the octave

Filters out keypoints that are edge points

Reduces impact of illumination changes on keypoints

Extrema Detection

Octave



Orientation Assignment

Magnitude and its orientation is calculated for all pixels around keypoint

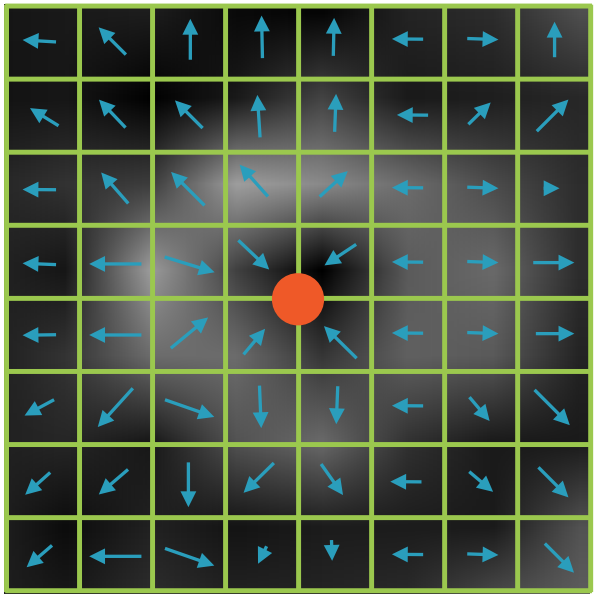
36 bin histogram is created to capture the orientations of all analyzed pixels

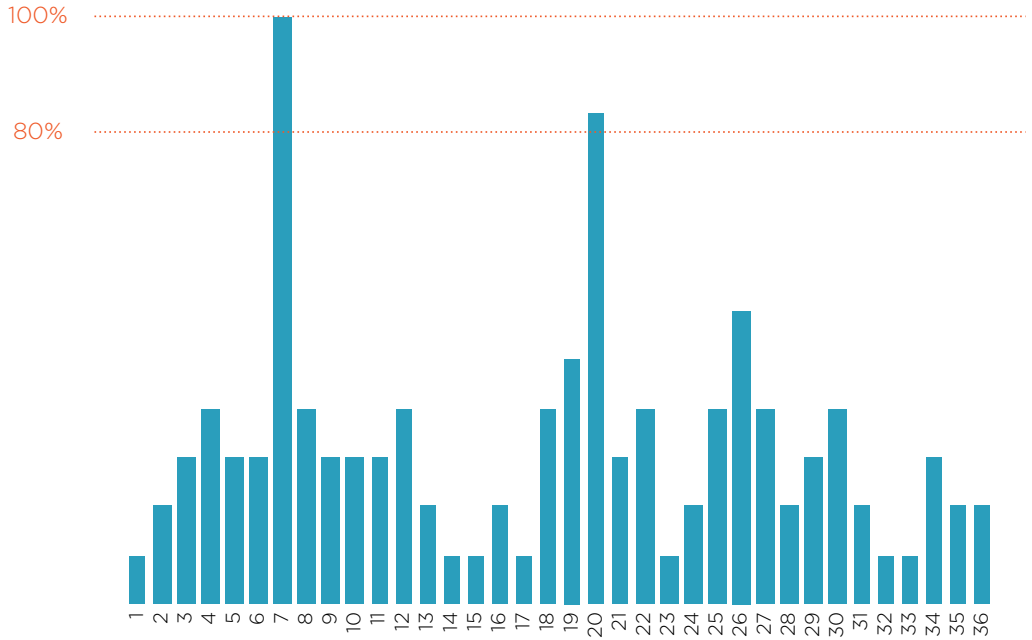
The histogram's highest point is the orientation for the keypoint

Any orientation above 80% of peak will be the orientation of a duplicate keypoint

Process enables keypoints to be rotation-invariant

Orientation Assignment





Keypoint Descriptor

The 16x16 block around the keypoint is considered for the final descriptor

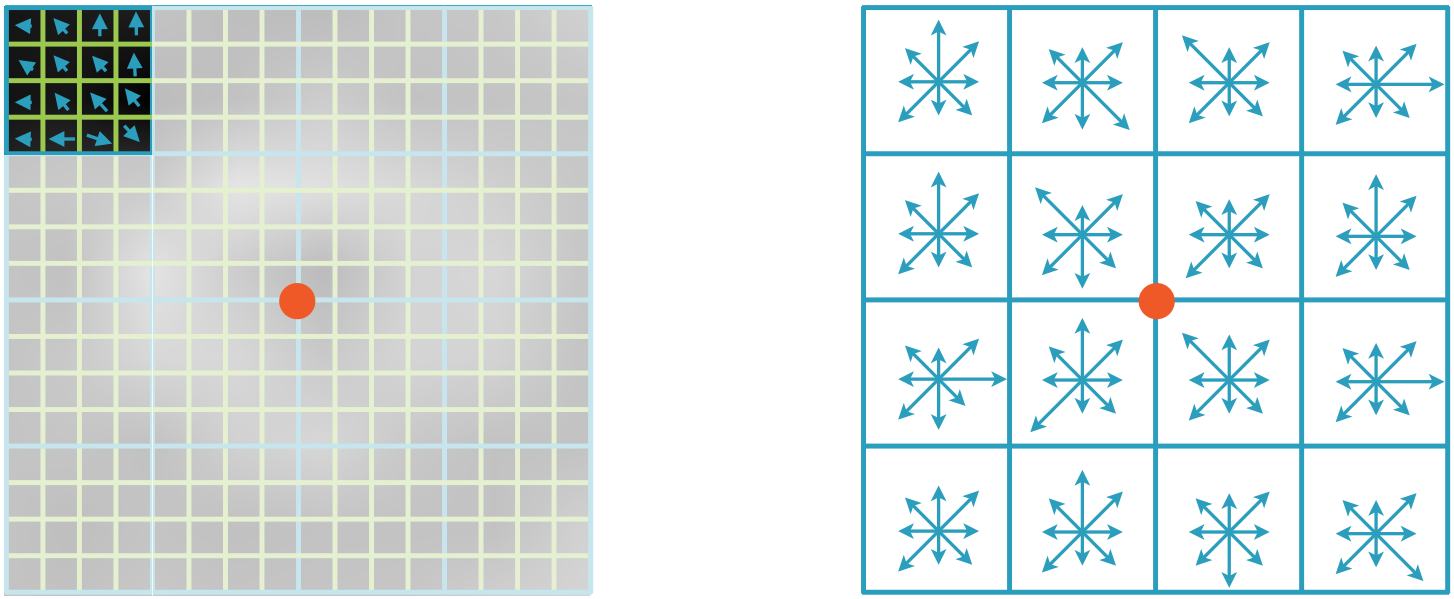
Orientation histograms are take in 4x4 pixel blocks around keypoint

Overall magnitudes have a Gaussian function applied

Modifications are made to the vector to minimize effects of illumination

Final descriptor is a distinctive 128 element vector

Keypoint Descriptor



8 bin histogram x **16** blocks = **128** element feature descriptor

Extracting & Matching Features with SIFT

Demo

- Launch the SIFT Jupyter notebook
- Install an OpenCV version that includes SIFT support
- Detect keypoints in an image using SIFT
- Review configuration options for SIFT detector
- Match keypoints across multiple images
- Verify rotation and scale-invariance with for SIFT keypoints

HOG Introduction

"The **histogram of oriented gradients** (HOG) is a feature descriptor used in computer vision and image processing for the purpose of object detection. The technique counts occurrences of gradient orientation in localized portions of an image."

Wikipedia

History of HOG

Robert K. McConnell

Introduced the concept of HOG in a patent application in 1986

Mitsubishi Electric Research

Utilized the concepts from McConnell in their work in 1994

Navneet Dalal & Bill Triggs

Presented pedestrian detection using HOG at CVPR in 2005

Person Detection

Because person detection was the use case that brought HOG back as a critical feature descriptor in computer vision, we will be leveraging this use case as we explore this algorithm.

HOG Algorithmic Steps

Gradient Computation

Orientation Binning

Block Normalization

Descriptor Calculation

Gradient Computation

Different implementations of HOG will have preprocessing prior to this step

Calculation of gradients accomplished by applying a horizontal and vertical kernel

Sobel edge detector can be leveraged instead of the default kernel

This phase outputs the magnitude and angle of the gradient at each pixel

Derivative Mask Kernel

$$\begin{bmatrix} 1 & 2 & 1 \end{bmatrix}$$

Kernel X

$$\begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix}$$

Kernel Y

Magnitude $\mathrm{g} = \sqrt{g_x^2 + g_y^2}$

Orientation $\theta = \arctan \frac{g_y}{g_x}$

Orientation Binning

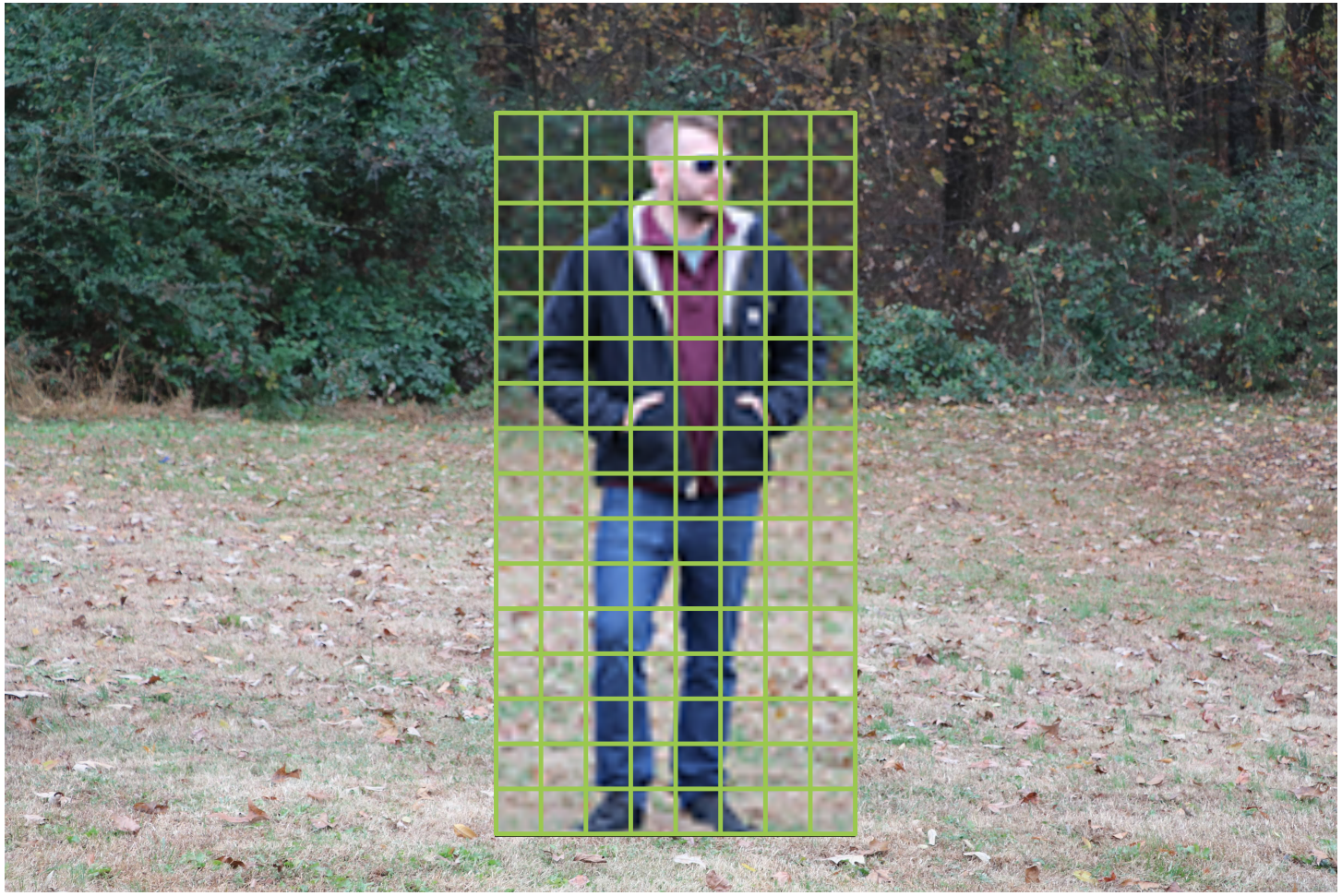
Utilizes 8x8 cells within the image

Each pixel's magnitude and orientation are considered

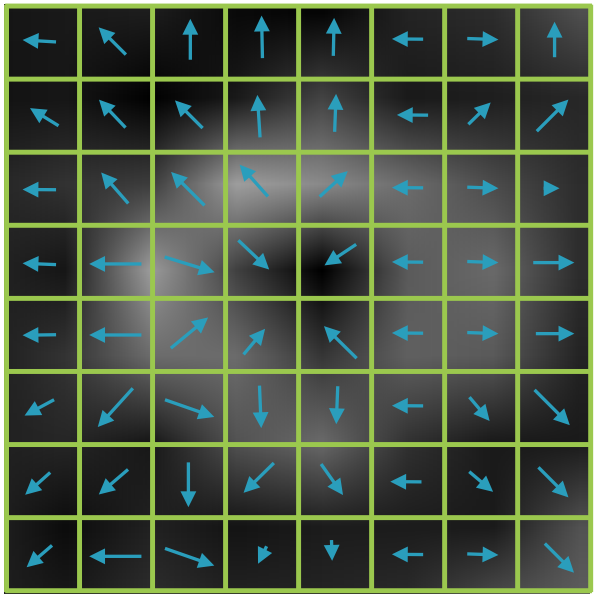
Results are placed in a weighted 9 bin histogram

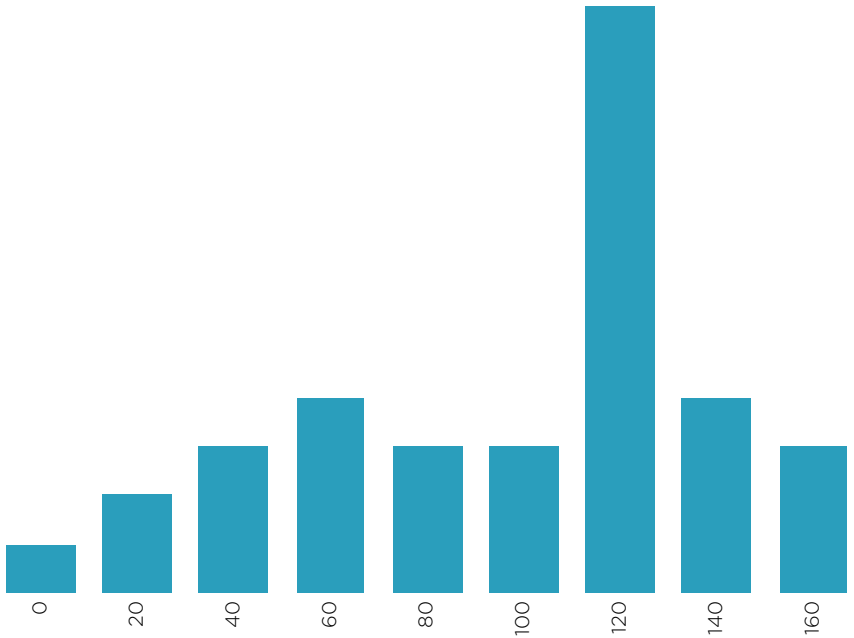
Different implementations support either signed or unsigned gradients

HOG Cells



Orientation Binning





Block Normalization

Utilizes larger 16x16 cells within the image

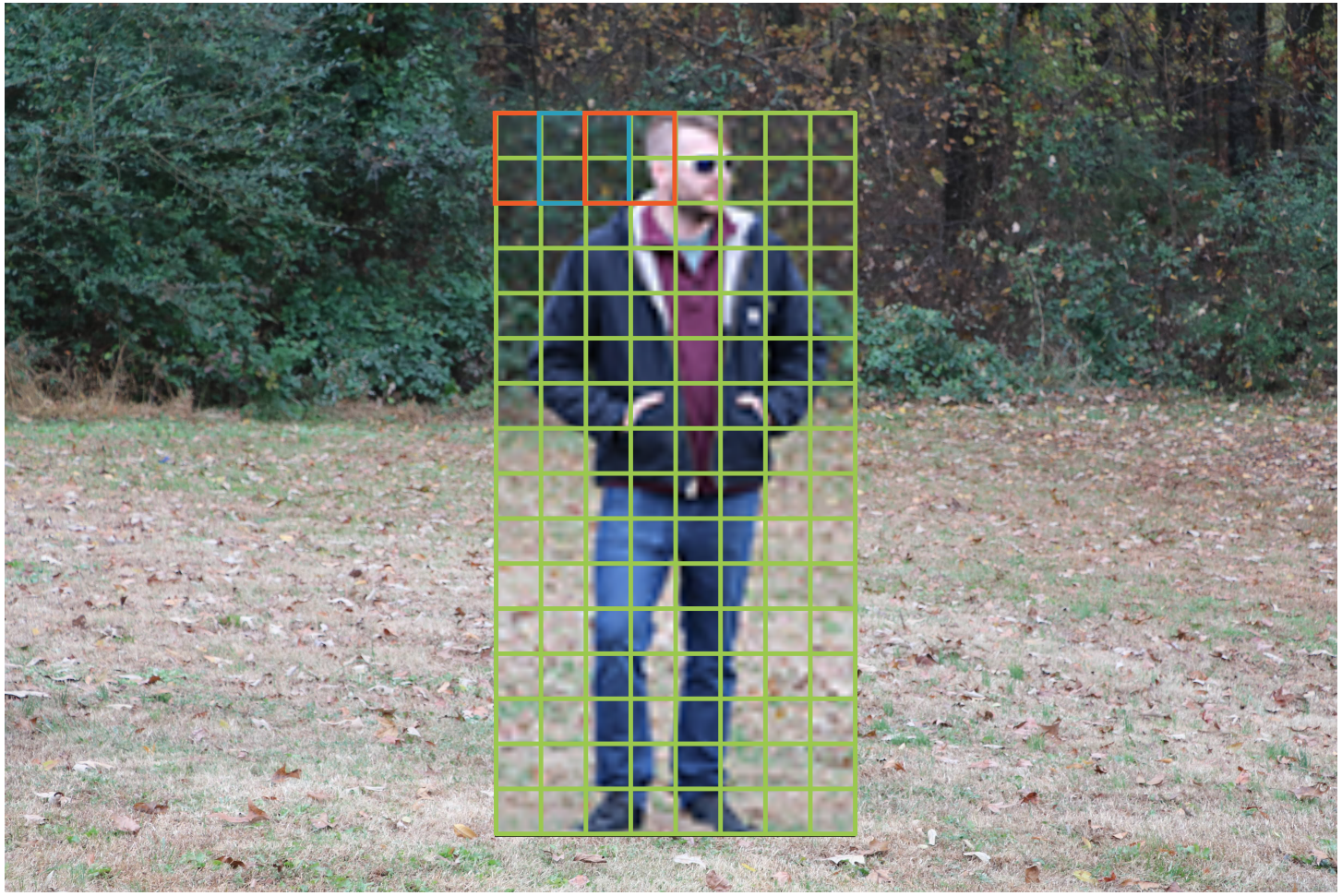
Blocks can be either rectangular or circular

Each will output a 36 element vector combining the 9 bin histograms of its cells

The vector is normalized based on the Euclidean norm

Other normalization approaches can be utilized

Block Normalization



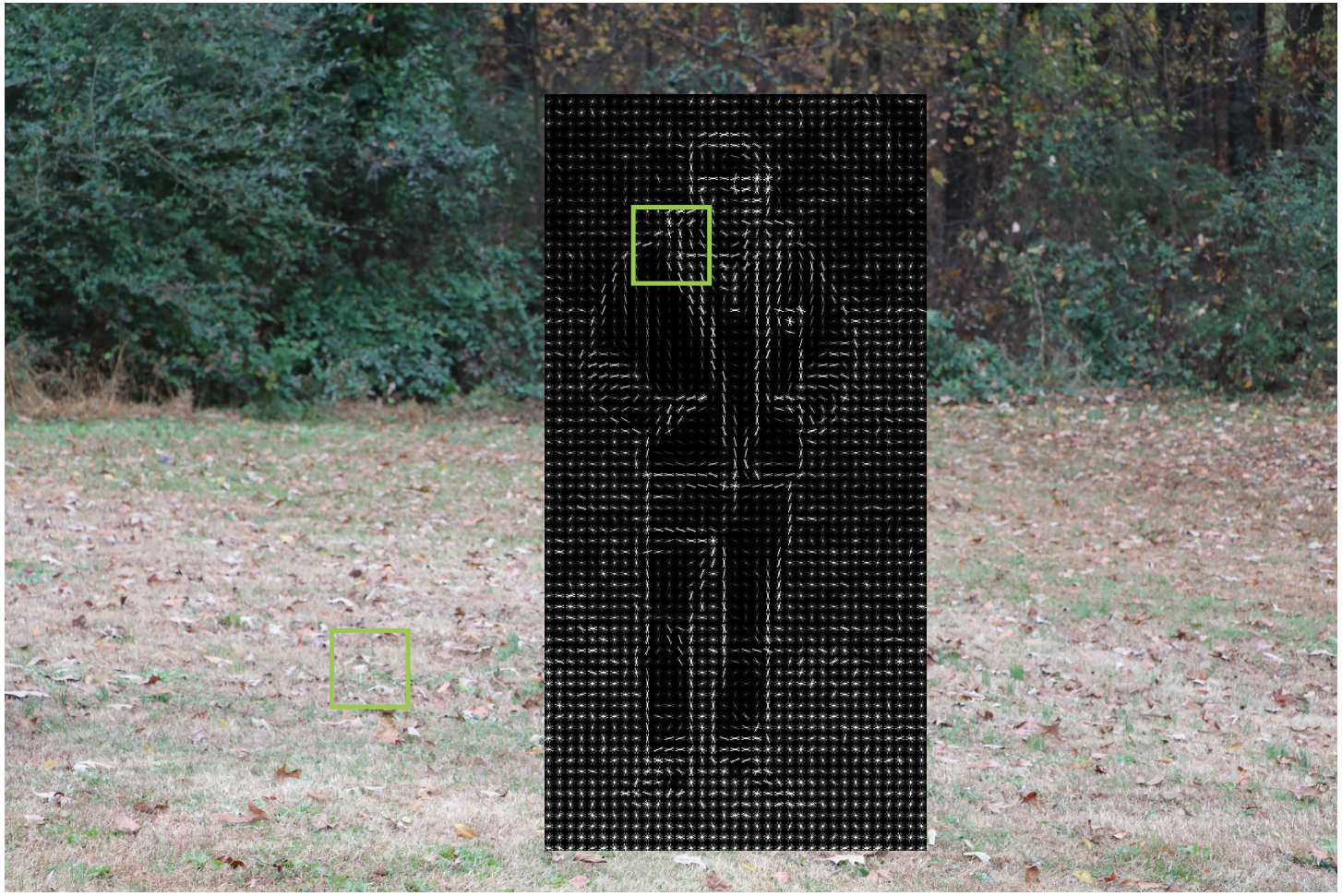
Descriptor Calculation

Each vector from the block normalization is an element of the feature vector

The block vectors are concatenated to form a single feature vector

For the person detector use case this will result in a 3780 element feature vector

Feature Vector



Extracting & Matching Features with HOG

Demo

Review needed modules to work with HOG

Create and visualize HOG vector for an image

Utilize OpenCV's integrated person detection with HOG

Summary

Summary

- Introduced image processing techniques
- Explored feature descriptors with the SIFT algorithm
- Utilized SIFT feature descriptors for matching
- Explored feature descriptors with the HOG algorithm
- Utilized HOG feature descriptors for matching