Context Free Application Development Plan

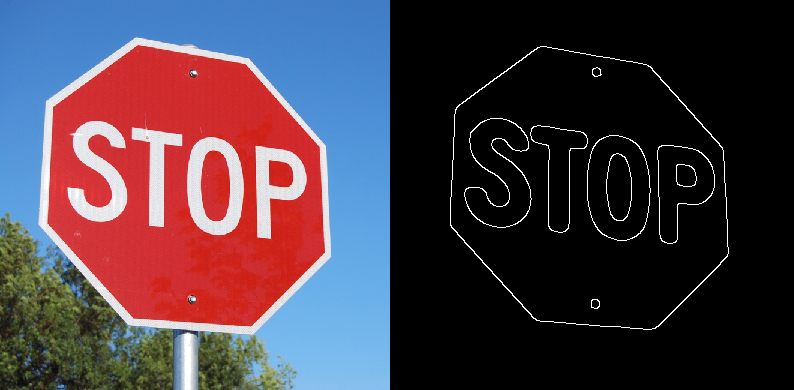


Figure 1: Canny Edge Detection run on a blurred version of (Red – (Green + Blue)) channel subtraction.

# Team

* Team name: Context Free
* Team members: Kevin Cherry, Robert Firth, Dennis Castleberry

# Requirements

* **Main goal**: to minimize the likelihood of pedestrian and driver casualties due to inaccuracies and inefficiencies in sign, crosswalk, and pedestrian detection algorithms.

To accomplish this, we propose implementing three algorithms for stop sign detection – one implementation of SURF and two new faster algorithms specifically devised for the task of fast stop sign detection. This algorithms are listed below as objectives.

* **Supplementary goals, objectives, and tasks**:
  1. **Goal**: stop sign detection. Detect stop signs in real-time with sufficient efficiency to allow for stopping time. Test SURF against two alternate algorithms, described below.
     1. **Objective**: construct an R-channel integral-image-based detector.
        1. **Task**: compute integral image on the R-channel of the image.
        2. **Task**: superimpose an*m*x*m*grid on the image and compute the sums of the R-channel intensity per 2x2 region.
        3. **Task**: for the 2x2 region with the maximum sum, recursively apply (ii) until the region with the maximum R-channel intensity density is obtained.
        4. **Task**: in the neighborhoods of the vertices inside the maximum-density region, find the four box such that the diagonal along those boxes separates a maximum-density R-channel triangle from a minimum-density R-channel triangle.
        5. **Task**: obtain the 8 vertices of the sign from the above.
     2. **Objective**: construct octagon detector.
        + **Task**: apply a shape detector to yield sets of vertices.
        + **Task**: discard all sets where *n*6= 8.
        + **Task**: check to see if the opposing edges of the 8-vertex shape are parallel (within a certain threshold).
  2. **Goal**: pedestrian crosswalk sign detection.
     1. **Objective**: construct and RG-channel integral-image-based detector.
        1. **Task**: compute integral image on the R-channel of the image.
        2. **Task**: superimpose an*m*x*m*grid on the image and compute the sums of the R-channel intensity per 2x2 region.
        3. **Task**: for the 2x2 region with the maximum sum, recursively apply (ii) until the region with the maximum R-channel intensity density is obtained.
        4. **Task**: in the neighborhoods of the vertices inside the maximum-density region, find the two boxes such that the diagonal along those boxes separates a maximum-density RG-channel triangle from a minimum-density RG-channel triangle.
        5. **Task**: obtain the 3 vertices of the sign from the above.

# Design

* **Modules** 
  1. **SignDetector1**:
* **Functionalities**: computes and uses integral image on the R-channel to find maximum density areas used to approximate the region occupied by the stop sign.
* **Data structures**:
  1. **Namespaces**: EmguCV.CV.
  2. **Classes** (Image) and **class members**: Resize(), Convert(),
* **SignDetector2**:
  1. **Functionalities**: uses edge detection, then isolates eight vertices with maximum octagonality.
  2. **Data structures**: Image, LineSegment2D, boxList, triangleList, Contour
  3. **Namespaces**: EmguCV.CV.
     1. **Classes** (Image, Contour) and **class members**: Canny(), HoughLinesBinary(), Resize(), Convert(), PyrUp(), PyrDown(), FindContours().
* **PedDetector**:
  1. **Functionalities**: detects pedestrians using built-in EmguCV classes.
  2. **Data structures**: Rectangle[].
     1. **Namespaces**: EmguCV.CV.
     2. **Classes** (Image, HOGDescriptor) and **class members**: Draw(), SetSVMDetector(), DetectMultiScale(), GetDefaultPeopleDetector().

# Implementation Plan and Timeline

• **Mod**

# User Interface

• **Mod**

# 

# Appendix

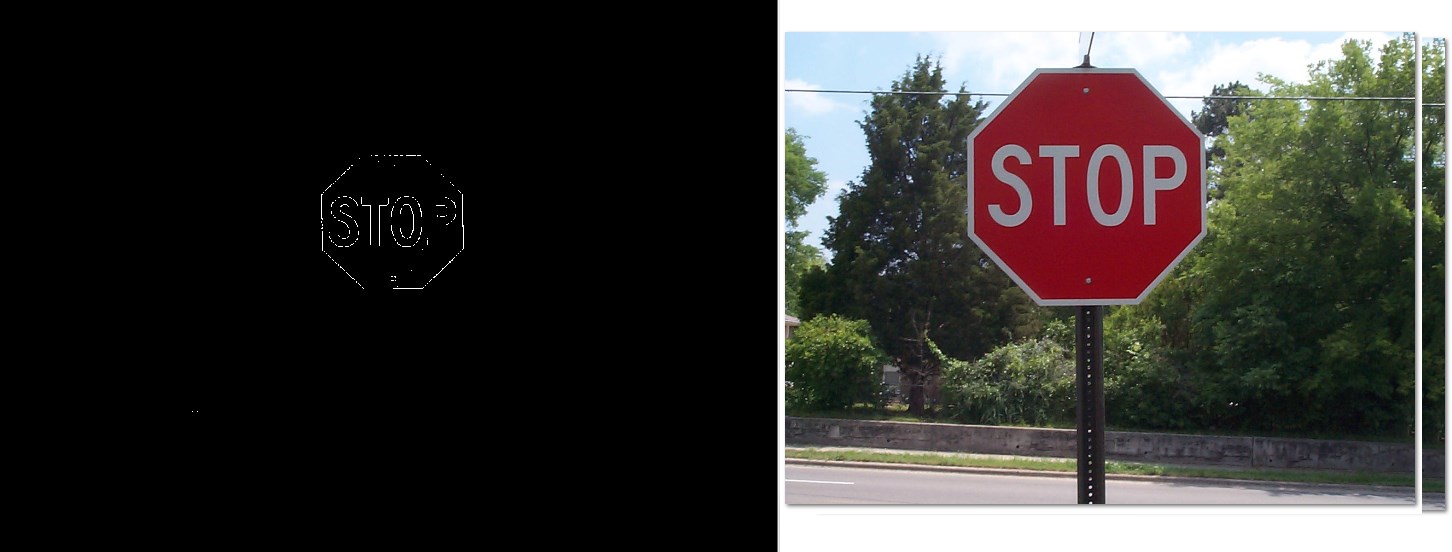


Figure 2: Edge detector by computing R-(G+B), clamped from 0 to 255



Figure 3: Edge detector by computing R-(G+B), clamped from 0 to 255