DATA DEPENDENT OPTIMIZATION VISION ARCHITECTURE

A Dissertation in
Computer Science and Engineering
by
Chris S. Lee

© 2016 Chris S. Lee

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

December 2016
The dissertation of Chris S. Lee was reviewed and approved* by the following:

Vijaykrishnan Narayanan  
Distinguished Professor of Computer Science and Engineering  
Dissertation Advisor, Chair of Committee

Jack Sampson  
Assistant Professor of Computer Science and Engineering

Sumeet Gupta  
Assistant Professor of Electrical Engineering

Soundar Kumara  
Professor of Industrial and Manufacturing Engineering

Kevin Irick  
Founder SiliconScapes, LLC  
Special Committee

Chita R. Das  
Distinguished Professor and Interim Department Head  
Computer Science and Engineering

*Signatures are on file in the Graduate School.
Abstract

One of the cardinal problems in computer vision is information overload. Image sensors generate a vast quantity of pixel data through multiple channels at high speed, and such a large amount of visual data is challenging to process in real time. Although a naive approach can reduce the visual information by re-sizing images, doing so results in a data loss. To limit the data loss, recently-proposed approaches leverage "regions of interest", which are an intelligently selected subset of samples in an image. Unfortunately, among techniques based on generating and consuming regions of interest, each shows different performance for different scenes, which makes one fail to achieve the best performance for a set of scenes with a single technique. Motivated by this, we propose a dynamic mechanism that selects the best technique (generating the best performance) depending on the incoming scenes.

In this dissertation, we propose several novel approaches to optimize computation
resources by exploiting natural redundancy, dynamic algorithm selection and application specific methods. In the first part of the dissertation, we present a hardware architecture that exploits natural redundancy across pixels, frames and channels. The architecture reuses and shares the results with minimized overhead to reduce power consumption. In the second part of the dissertation, we propose a dynamic saliency algorithm selection technique that is able to choose the best saliency map based on machine learning. We demonstrate the benefits of the approach using a sampling approach across video frames. Thirdly, we show how to apply saliency maps to grocery scenes to address challenges arising for object class detection in real world scenarios.
# Table of Contents

List of Figures vii  
List of Tables x  
Acknowledgments xi  

## Chapter 1  
Introduction 1  
1.1 Exploiting Natural Redundancy in Visual Information 2  
1.2 Dynamic Prediction of Per-Input Saliency Algorithm Preference 3  
1.3 Attention-Seeded Opportunistic Template-matching 4  
1.4 Outline 4  

## Chapter 2  
Exploiting Natural Redundancy in Visual Information 6  
2.1 A Neuro-Inspired Vision Pipeline 8  
2.1.1 Retina Preprocessing 9  
2.1.2 LGN Mixing 9  
2.1.3 Feature Extraction 10  
2.1.4 Salient Region Detection 11  
2.1.5 Object Detection and Classification 11  
2.2 Redundancy in Visual Information 12  
2.2.1 Spatial Similarity Within a Frame 12  
2.2.2 Channel Redundancy 15  
2.2.3 Temporospatial Redundancy 17  
2.3 Exploiting Input Redundancy in A Neuro-Inspired Vision Pipeline 20  
2.3.1 Overall Optimization Architecture 20  
2.3.2 Hardware implementation 21  
2.4 Experimental Results 23  
2.5 Related work 28
2.6 Sub Conclusion ................................................. 31

Chapter 3
Dynamic Prediction of Per-Input Saliency Algorithm Preference 32

3.1 Related Work .................................................. 34
  3.1.1 Selective Search for Object Recognition .................... 34
  3.1.2 Learning-based Salient object detection ..................... 34
  3.1.3 Salient object detection ranking ........................... 35

3.2 Motivation ...................................................... 35

3.3 Approach of this work ........................................ 37
  3.3.1 Saliency Map .............................................. 38
  3.3.2 CNN Training ............................................ 39

3.4 Experimental Evaluation ..................................... 40
  3.4.1 Evaluation ............................................... 40

3.5 Sub Conclusion & Discussion .................................. 42

Chapter 4
Attention-Seeded Opportunistic Template-matching 47

4.1 Related Work .................................................. 50

4.2 Detection and Classification using Attention-Seeded Opportunistic
  Template-matching .............................................. 51
  4.2.1 ASOT-E(xhaustive) ....................................... 51
    4.2.1.1 Template Matching Phase ............................ 53
  4.2.2 ASOT-O(pportunistic) .................................. 56
  4.2.3 ASOT-S(cheduled) ....................................... 58

4.3 Experimental Results ......................................... 59
  4.3.1 Evaluation on the GroZi-120 dataset ....................... 60
  4.3.2 Impact of class repetition on performance ................ 61

4.4 Discussion .................................................... 63

Chapter 5
Conclusion ....................................................... 68

Bibliography ...................................................... 71
List of Figures

2.1 Neuro-Inspired Vision Pipeline. ........................................... 8
2.2 Pseudo Code for Similarity Filter. ....................................... 13
2.3 Similarity Redundancy in 3x3 Sliding Window. ......................... 14
2.4 Opportunities for redundancy exist across channels (R, G, B). ....... 16
2.5 Architecture for exploiting channel redundancy. ......................... 17
2.6 Architecture for exploiting Temporospatial Redundancy. ............... 18
2.7 Temporospatial Redundancy in Video. .................................... 18
2.8 Redundancy Rate. .......................................................... 20
2.9 System architecture that combines exploitation of spatial, channel and temporospatial redundancies. ........................................... 22
2.10 Gabor filter representing one channel in the feature extractor. ....... 22
2.11 Buffer & Multiplier Set. .................................................. 24
2.12 Reduction in computations due to exploiting redundancy for Feature extractor for a sample video. .............................................. 26
2.13 Influence on Error Rate on the Gabor Feature extractor. ............... 26
2.14 Variation in computation reduction for different thresholds for the feature extractor averaged across all videos for moving and fixed cameras. .......................................................... 27
2.15 Gabor Feature extractor Error Rate. ...................................... 28
2.16 Power Consumption for Gabor Feature Extractor. ....................... 29

3.1 General structure for object detection and classification. ROI proposal provides location of object in the given image. Feature extraction optimizes information of image to analyze it faster in next step. Classifier discovers what ROI is. .................................... 36
3.2 Saliency detection examples. Input (a) and (i) has same target object which is foreground region on Ground Trues (e) and (m). Another figures are results of these input by 6 different saliency detection algorithm. Although two input scene has same target object, these results are different by algorithm. .................. 43
3.3 Examples to show performance based on weighted f-measurement. Images on top line are six sampled input frames in ILSVRC-2015 VID data-set. Second line is their ground trues. Third line and bottom line show thresholded saliency map from SWD and FES. The graph shows alterations of performance based on weighted f-measurement. These two lines are similar shapes. Performance of SWD is better than FES for most of frames. However, these performances are switched only between frame 74 and frame 101.

3.4 Dual saliency detection with predictor. Saliency predictor finds that which saliency detection method is better base on a given image. Saliency predictor provides enable signals and selection signals for two saliency detection methods and multiplexer.

3.5 Evaluation of FES and SWD based on AUC score. Blue dots indicate 60,000 of input images. Blue dots are on dashed line when the scores of two methods are same. Solid line which gives 10% advantage to FES is threshold to decide labels for training. Images which are relevant blue dots over the solid line are labeled for SWD. Others are labeled for FES.

3.6 Accuracy of prediction. The predictor selects correct method for 65% of test images.

3.7 Comparison of AUC Scores. Green line indicates AUC score of SWD (0.738). And Blue line is AUC score of FES (0.6777). These are straight lines to show their average AUC scores in the testing. Red line shows alteration of this method. This performance is slightly higher than SWD when it uses lower than 100 sampling period.

3.8 Computation time with prediction. Green line is the running time of SWD (2.33s). And blue line is the running time of FES (0.41s). These are straight lines to show their average running times in the testing. Red line is the running time when CNN is processed on CPU. Black line shows the running time when CNN is processed on GPU.

4.1 Examples of repeated instances of a class in visual inputs

4.2 ASOT vision-processing pipeline, including RoI scheduling framework

4.3 The left image shows pairs of matched features. A transformation matrix T is calculated by matching features with the best template image. The image on the right depicts the bounding box of the template translated and rotated into image space.
4.4 The initial(left) figure shows the seed object(green) and matched features(red). These matches are described by a 2D histogram(middle) of \( tr^x \) and \( tr^y \) for matched features. The histograms are then filtered for noise(right). To reduce noise, this work used a Gaussian filter and thresholds.  

4.5 Output and recall/time performance of ASOT-E,-O, and -S on example image  

4.6 ASOT Retrieval Examples (1)  

4.7 ASOT Retrieval Examples (2)
List of Tables

2.1 Redundant windows, by similarity threshold. . . . . . . . . . . . . . 15
2.2 Channel redundancy, by similarity threshold. . . . . . . . . . . . . . 17
2.3 Temporo-spatial redundancy, by similarity threshold. . . . . . . . 19
2.4 MEMORY & FLAG. . . . . . . . . . . . . . . . . . . . . . . . . . . 23

4.1 Performance of saliency map algorithms on Dataset for this work. . 59
4.2 Performance on Grozi-120. . . . . . . . . . . . . . . . . . . . . . . . . 60
4.3 ASOT performance on supermarket dataset. . . . . . . . . . . . . . 61
4.4 Time to reach specified fraction of recall saturation. . . . . . . . . 61
4.5 Performance of ASOT-E on the Dataset. . . . . . . . . . . . . . . . 61
4.6 Multi-template classes in in-house grocery dataset . . . . . . . . . 63
Acknowledgments

This dissertation would not have been possible without the help of many people. I want to thank foremost my advisor Prof. Vijaykrishnan Narayanan for his constant support and guidance during the development of this work. He suggested new ideas for me to explore and have helped me appreciate the principles of scientific work.

I also would like to thank Prof. Jack Sampson who gave me generously detailed advices for how to approach scientific problem and challenges. I also want to thank my committee member Prof. Soundar Kumara and Sumeet Gupta. It was a honor to have a great committee. Many thanks to Dr. Kevin Irick. He suggested many good research topics to work in MDL. To all the MDLers including Ahmed, Sungho, Karthik, Matt, Yang, Misun, Siddharth, Nandhini, Huichu, Moonseok, Jinhang, William and Yasuki : It was great to meet and work with all of you. In particular, I want to thank my friends Shawn, Manho, Yungmin and Kyungrin.

Finally, I give my deepest gratitude to my family. I am proud of my wonderful
family who have always supported me in the pursuit of further education.
Chapter 1

Introduction

Computer vision systems are ubiquitous due to the proliferation of devices equipped with a camera such as smart phones, smart watches, drones and wearable devices. It provides the infrastructure to develop a smart vision system that creates a wide range of new functionalities. The image processing techniques are also applied in everyday life, not only in the special services such as medical images, military images and microscope images. Smart vision provides associated information when the camera shoots only the products, buildings, landscapes, faces, or other images. The high lever recognition that resembles human vision needs complex computations with large amount of data to account for the various cases. This dissertation presents methods to optimize computer vision for mobile systems in the following three scopes: (1) power deduction exploiting natural redundancy in visual information and (2) appropriate algorithm selection for target scenes (3) optimization of computation by repeated input search.
1.1 Exploiting Natural Redundancy in Visual Information

Visual information consists of multiple input channels such as RGB, HIS and stereo vision. To process complex vision tasks, there are two critical problems which are battery life and CPU performance. To process these in real-time, the system needs multiple hardware pipelines with high-throughput. In general, vision accelerators are designed to optimize power consumption and hardware space.

One of the common techniques is to approximate computation that is considered in design-time. Another technique is the hardware accelerators trade off accuracy versus efficiency on runtime. For example, if the input scene is the blue sky, almost all of the pixel values will be the blue color that is 0, 0 and 255 in the case of 8-bit RGB channel. We can anticipate the entire result using only one computation instead of the full analysis. In particular, video frames are often redundant in adjacent pixels, across frames and channels. Our proposed method focuses primarily on dynamic power reduction. We use clock and value gating techniques similar to Value-Based Clock Gating and Operation Packing [6] that was published by Brooks in 2000, although with a very different policy. Other power-saving proposals are reduction leakage and dynamic power consumption [7] by controlling power supply that was introduced by Yoshimi. Window memoization [8] has been proposed as an efficient method for high performance implementation of image processing. Unlike
[8], we focus on dynamic power deduction, rather than speed-up, and our technique does not require large memorization tables, as we only seek to exploit local value consistency that can reduce power via eliminating transitions.

1.2 Dynamic Prediction of Per-Input Saliency Algorithm Preference

There is a wealth of image processing and visual computing algorithms occupying different trade offs between accuracy and complexity/cost metrics. To further complicate matters, while there is often a strong general correlation between implementation/execution cost and task accuracy across entire input sets, for a given input or input class, a cheaper approach will sometimes provide superior accuracy. In this paper, we explore the predictability of input-to-technique affinity when balancing between execution time and accuracy. We show that a neural network predictor can be trained to detect scenarios where a less computationally intense saliency algorithm is likely to produce the same or similar Area-Under-Curve(AUC) results to a more expensive algorithm. We demonstrate that using this predictor with periodic sampling of a video stream can simultaneously improve AUC for region detection over both baseline saliency models (0.743 versus 0.738 and 0.678) and decrease running time by employing the cheaper saliency algorithm in up to 36% of frames.
1.3 Attention-Seeded Opportunistic Template-matching

The retail space is rich with both applications and challenges for computer vision tasks, such as object detection and classification. While the potential benefits from visual-assist and other embedded applications in these environments are clear, the retail domain can feature large numbers of object classes, diverse, non-planar object geometry, and collections of objects from similar classes in close proximity. Moreover, real-world video inputs to embedded systems exhibit significant variance in lighting, viewing angle, and distance, further complicating visual tasks. We introduce a new visual detection system, Attention-Seeded Opportunistic Template-matching (ASOT), that is tailored to deal with two key visual aspects of the retail environment. Namely, ASOT is optimized for an expectation of repeated instances of a given class in a single image, and handles the unstructured alignment and rotations of objects in real-world retail inputs. We evaluate ASOT on the GroZi-120 data-set and show improvements in recall and precision over prior work. We then use an internally developed grocery data-set to explore ROI scheduling optimization for images that contain many instances of the same object.

1.4 Outline

The remainder of this dissertation is structured as follows. Chapter 2 presents three types of redundancy in visual information and describes a hardware implementation.
to exploit input redundancy in neuro-inspired vision pipeline. Chapter 3 explains dynamic prediction to choose the best algorithm on run-time. Chapter 4 discusses reusable information for object detection. Chapter 5 is the conclusion.
Chapter 2  
Exploiting Natural Redundancy in Visual Information

Increasingly complex mobile devices, such as smart phones, are now ubiquitous. Such devices often feature cameras, making portable photo and video capture facilities equally pervasive. These cameras operate at relatively high resolutions, and can produce large amounts of high-quality visual data very rapidly. Even low power, wearable devices, such as [1] are viable sensor platforms for feeding computer vision systems tasked with complex tasks including object recognition, tracking, and scene categorization. However, to perform the necessary analysis on the visual data, there are both performance and energy hurdles to overcome in either executing the analysis locally or transmitting the data for remote analysis. From an energy-efficiency standpoint, it is preferable to maximize the degree of local analysis performed. Simultaneously, in order to keep up with the incoming frame rate within
the form-factor (thermal) and power (battery-life) constraints of an embedded
system, the amount of computation performed per frame must be limited, even in
dedicated accelerator pipelines. Neuro-inspired vision systems address both power
and performance requirements, and, like their biological counterparts, are inherently
robust to the incomplete and noisy visual stimuli that they typically process. Neuro-
inspired algorithms for computer vision, and the hardware accelerators that embody
them, trade off accuracy versus efficiency via exploiting the non-uniform nature
of information density in visual data. Through pre-processing heuristics, such
as salience [2] only subsets of the overall images are ever considered in detail.
However, current design-time optimized accelerators, while exploiting one aspect of
informational non-uniformity, do not exploit other forms of informational asymmetry
that are common in visual data. In particular, visual data is often redundant in
space, time, or both. Further, even where portions of an image near in space or time
are not identical, they are often similar, and of similar importance to the overall
output of the visual analysis algorithm. Since the algorithms are already robust to
incomplete noisy inputs, portions of the input deemed redundant can be suppressed
without necessarily degrading output quality. Thus, there is an opportunity to
exploit dynamically tunable notions of similarity for visual inputs when suppressing
computation. This paper presents three methods to reduce three different aspect
of input-dependent redundant computation. Section 2.1 describes the baseline
neuromorphic vision pipeline. Section 2.2 describes methods for exploiting spatial
redundancy within a frame, redundancy across channels representing components of the frame data, and temporospatial redundancy in video input. Section 2.3 presents a modified version of the neuromorphic vision pipeline that incorporates the three techniques described in Section 2.4 evaluates the efficiency and accuracy trade-offs of approach of this work. Section 2.5 discusses related work, and Section 2.6 sub concludes.

2.1 A Neuro-Inspired Vision Pipeline

Figure 2.1 illustrates an embedded neuro-inspired object detection and classification system based on the two streams model of biological vision. The system utilizes streaming hardware accelerators to perform several of the early preprocessing and feature extraction stages. These hardware accelerators exhibit significant energy efficiency beyond that of an equivalent embedded software implementation. The following subsections describe the functionality of each of the subcomponents of the vision system.

![Figure 2.1: Neuro-Inspired Vision Pipeline.](image-url)
2.1.1 Retina Preprocessing

The Retina accelerator performs localized contrast enhancement and dynamic range compression of the input imagery [3, 4]. The primary components of the Retina pipeline are streaming Center-Surround Differencing, CSD, and Non-Linear Activation, NLA, operators. The CSD operator performs local contrast enhancement by pixel-wise amplification of the difference between the intensity of a center pixel and the weighted aggregate of the surrounding pixel intensities. The NLA operator performs dynamic range compression by normalizing each enhanced pixel using a steep-slope sigmoid function modulated by the surrounding pixel statistics.

2.1.2 LGN Mixing

The LGN accelerator performs mixing of the retina enhanced image channels in a way that amplifies either the mutual or opponent information content [3]. The mixing allows additional input modalities such as SWIR and LWIR imagery to be combined with standard luminance and color channels while constraining the output to a manageable and fixed number of channels. The LGN accelerator implements the inter-channel mixing using Center-Surround Differencing and Non-Linear Activation in a fashion similar to the contrast enhancement and normalization processes performed in the Retina accelerator. The LGN accelerator accepts five contrast enhanced channels from the upstream retina process that includes Y, I, Q,
Negated I, and Negated Q channels. Subsequently, the LGN accelerator produces four fused channels that are used in the downstream feature extraction stage.

### 2.1.3 Feature Extraction

Before object detection and classification is performed, the enhanced input imagery is transformed into a feature representation. Each channel produced by the LGN stage is projected into a Gabor feature representation by convolution with several 2D Gabor filters. Fundamentally, 2D Gabor filters are orientation and frequency selective edge detectors that effectively approximate the behavior of complex cells in the early visual pathway of primates. A specific Gabor filter can be tuned to produce its peak response when induced by an input gradient of a particular spatial orientation (angle) and extent (scale). By applying several Gabor filters at varying orientations and scales the input image can be reduced to a sparse representation that can lead to more robust and computationally efficient object detection and classification. In total, 144 convolutions are performed across four fused channels, six Gabor orientations, and six spatial extents (kernel sizes). The resulting Gabor feature maps are subsequently used in the attention and classification stages. Although, each Gabor filter is orientation and scale selective, they exhibit decreasing non-zero response to neighboring orientations and scales. This redundancy can either by removed by a pooling operation or leveraged to increase the robustness of some downstream perception tasks.
2.1.4 Salient Region Detection

The salient region detection accelerator determines the location of regions-of-interest based on a feature measure of "pop out". The accelerator is based on the AIM model proposed by Bruce and Tsotsos [2]. Using a center surround scheme the AIM algorithm assigns a likelihood value to each pixel based on the probability of finding that pixel’s value in the local surround. The resulting information map is thresholded and low-pass filtered to produce a saliency map. A connected component process then assigns bounding boxes to the saliency map. These bounding boxes are the image locations for which the subsequent object classification is performed. The object classification accelerator assigns a label to objects that it successfully detects in the regions-of-interest provided by the saliency detector. This accelerator is based on the HMAX feature pooling and template matching algorithm. The algorithm extracts high-order V2 cortical features from the V1 features that were produced in the upstream feature extraction stage. Final classification of the resulting high-order cortical features is performed by an appropriately trained Support Vector Machine.

2.1.5 Object Detection and Classification

Object classification is the process of assigning a categorical label to an unknown object. Semantic reasoning about a scene can be achieved by identifying the objects
that comprise the scene and their relation to each other. Exhaustively classifying each patch in an image is a computationally prohibitive proposition, and so, object detection is employed to filter the image patches to those likely to contain an object. One type of object detection is based on the concept that an image patch worth consideration is one that is unlikely to appear in the current context. For example, the appearance of a single reddish circular object in the context of a green pasture exhibits high "pop-out" or more formally saliency. A salience value can be assigned to each x, y location in an image by computing the pixel’s likelihood of appearance with respect to some context. The context may be defined as the entire image, or some other region of semantic relevance. Clusters of peak salience values are an indication of an image region worth consideration by the classification system. Saliency is first computed for each preprocessed channel independently. Finally, a single saliency map is computed by pixel-wise aggregation of the individual channel saliency maps.

2.2 Redundancy in Visual Information

2.2.1 Spatial Similarity Within a Frame

Intuitively, it is easy to imagine sources of spatial redundancy within an image. For example, in an outdoor setting much of the upper portion of an image may be taken up by a blue sky. While later stages of the pipeline may suppress these portions
of the image through techniques such as salience (or background identification in traditional vision systems), significant resources are consumed in the pre-processing stages and feature extractor stage for such portions of the image as well. To exploit this, the nature of both the representation of the data and the streaming nature of current visual accelerator pipelines require some subtlety in exploiting this form of redundancy.

![Figure 2.2: Pseudo Code for Similarity Filter.](image)

Consider a 3x3 sliding window over the pixels of an image, as shown in Figure 2.3 (a). While consecutive windows may be very similar, the larger the window, the less likely they are to be completely identical. In Figure 2.3 (b), this work proposes to augment the sliding window with a threshold-tunable filter, the pseudo code for which is in Figure 2.2. This input-filter only changes output when the new streaming input pixel is more than the threshold value away from the last latched input pixel, and latches inputs when it changes output values. Thus, it
provides a stable output for runs of similar values, but will transition at line rate for edges, noise, or other steep gradients. Table 2.1 shows the average rate of window uniformity under different thresholds for uniformity across 30 test videos. If this work considers only identical computations, the opportunity for improvement is quite limited, but if this work allows for the elision of similar, as well as identical, regions, significant fractions of computation can be avoided.
In the 3x3 sliding window example illustrated in Figure 2.3, the system needs to monitor 9 input values for a 3x3 window to find a similar matrix, either exact match or partial match after the thresholding transformation. The partial matches can selectively reuse parts of the computation. Clearly, in order for this technique to be profitable, the elided computation must have greater complexity or resource investment than the filtering and similarity comparison. Similarly, by discarding data, the system could potentially reduce in accuracy. This work discusses the hardware needed to efficiently to skip uniform inputs in Section 4, and show the resource/power/accuracy trade offs in Section 5.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>28.5%</td>
<td>39.1%</td>
<td>51.04%</td>
<td>63.03%</td>
<td>74.27%</td>
</tr>
</tbody>
</table>

### 2.2.2 Channel Redundancy

As part of emulating the primate visual system, many neuromorphic vision pipelines decompose the incoming sensor data for a single frame into separate channels such as RGB, YIQ and HSI. This is done to extract channel specific information utilized to perform image enhancements (color correction) or extract task specific information (such as find a pink apple instead of a yellow banana). Some of these channels have very high similarity with other channels, as seen in Figure 2.4. The relative difference between pixels across channels remains constant (or within a threshold).
Such regions are defined as uniform regions. For linear operators commonly used in this pipeline stages, this similarity also applies to the output. Consequently, one of the pipelines (See Figure 2.5) is fed the differential input and the output is similarly reconstructed combining the output from both channels. This approach, when combined with the similarity filter proposed within a frame in the previous sub-section, can enhance the potential savings.

Figure 2.4: Opportunities for redundancy exist across channels (R, G, B).

Table 2.2 shows the rate of channel redundancy across the similarity filter thresholds for the test videos. The opportunity to remove redundancy ranges up to 82%.
2.2.3 Temporospatial Redundancy

In video, the input from the camera sensor per frame can be similar across the frames. For example, if background does not move between frames, the visual system can reuse partial results of the previous frame. Moreover, as this work has shown in the previous two subsections, a relaxed notion of similarity allows for substantially greater returns. In the above example, if the background was not stationary, but merely almost stationary, the accuracy of the visual system is still unlikely to suffer greatly if the output from the previous processing of the background region is reused. This work calls this "approximate memoization".

Figure 2.6 illustrates the hardware structure to reduce temporospatial redundancy across the frames in the video. This structure stores one sampled input frame (SI) to subtract target frames. This structure processes the differential of
the input frame and SI. This structure subtracts the result and sampled output (SO) to recover the original output. If the system can optimize the computation over approximately uniform regions, the system can save processing resources.

Figure 2.7 shows absolute differential of frames in our sample video. Adjacent
frames have more uniform regions than those frames further apart. For example, when the system reuses the result of frame 1 for frame 2, the accuracy will be higher than reusing the frame 1 results for frame 8. The system needs to update SI and SO when the temporospatial redundancy is small between current frame and SI. However, if this work updates SI and SO frequently, the overhead will be increased. This structure needs memory space to store SI and SO which is the same size of a frame. Further, two subtraction modules are necessary to calculate differentials. To balance the trade off between accuracy and overheads, the system has to have a strategy to identify the update frequency of SI and SO.

Table 2.3: Temporo-spatial redundancy, by similarity threshold.

<table>
<thead>
<tr>
<th>Threshold</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>4</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Redundancy</td>
<td>63%</td>
<td>72.1%</td>
<td>78.2%</td>
<td>81%</td>
<td>82.6%</td>
</tr>
</tbody>
</table>

Table 2.3 shows the average rate of temporospatial redundancy between adjacent frames using similarity filter in 30 videos. The opportunity for redundant computation elimination increases to 83% for a threshold of 8.

Figure 2.8 illustrates the average rate of the three redundancies in the 30 videos. The redundancy within a frame is already substantially reduced once this work applies the similarity filter with a threshold greater than one. Both temporospatial redundancy and channel redundancy approaches build upon this spatial redundancy to achieve even higher opportunities for exploiting redundancy. Our goal is to exploit the mutually complementary techniques of channel and temporal redundancy.
2.3 Exploiting Input Redundancy in A Neuro-Inspired Vision Pipeline

2.3.1 Overall Optimization Architecture

Figure 2.9 shows the overall structure for a two channel vision pipeline that combines the exploitation of the three sources of redundancy. A similarity filter is used for both the channel redundancy path and temporospatial redundancy path. The structure can permit exploiting either temporospatial or channel redundancy for each channel along with spatial redundancy.
and always engages intra-frame spatial redundancy. Since the temporospatial redundancy provides the maximum reduction in computation, this is the preferred option. However, the system has to update SI and SO when the input frame is highly different from SI. Such instances provide the opportunity for exploiting channel redundancy. As an example, consider four channels. For frame 1, channels 1 and 3 exploit only the intra-frame spatial redundancy engaging the similarity filter. Channels 2 and 4 exploit the channel redundancy from computations performed in channels 1 and 3 respectively (in addition to the similarity filter for intra-frame spatial redundancy) for frame 1. For frame 2, all channels can use the temporospatial redundancy in addition to the intra-frame spatial redundancy. They continue in this mode until the reference frame becomes outdated and needs to be updated. The system dynamically determines the need for update based on the decrease in the uniformity region from the previous frame. If the uniformity region decreases by 5%, this work updates SI and SO and the channels behave similar to frame 1 in this case. Section 5 quantifies the rate of these policy changes.

2.3.2 Hardware implementation

In order to illustrate the implementation details of our approach, this work focuses on the design of a feature extraction module in our pipeline. Figure 2.10 shows a single channel computation of the Gabor feature extractor pipeline augmented with the similarity filter. This structure operates on a 19x19 size image window
and utilizes gated clocks in the uniform regions to reduce computational power. This reduces the power in the memory modules, multiplier array and adder tree exploiting the redundancies.

The memory module in the pipeline stores only values different from the previous input (see Table 2.4). It should be observed that the probability of the match increases as the threshold of the previous stage similarity filter increases. This memory needs a (1-bit x image width) flag buffer to indicate the value is inferred when reading by the output of the previous cycle in this streaming architecture.
Consequently both read power and write power are saved using this memory structure.

<table>
<thead>
<tr>
<th>Table 2.4: MEMORY &amp; FLAG.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
</tr>
<tr>
<td>1</td>
</tr>
<tr>
<td>---</td>
</tr>
<tr>
<td>Input Data</td>
</tr>
<tr>
<td>Memory</td>
</tr>
<tr>
<td>Flag</td>
</tr>
</tbody>
</table>

The invert of the flag bit can also be used to gate the computations for the multipliers (and the adder chain not depicted in picture), since the output of the computations are exactly the same for this pixel as the previous one. In addition to the pixels, the shift registers also shift the flag bits to avoid additional comparisons.

Similar approach is used in other structures of our pipeline to exploit these redundancies.

### 2.4 Experimental Results

This section tested 30 videos, comprising 10 indoor and 20 outdoor videos, to evaluate our methods. The videos included 17 using moving cameras and 13 using still cameras. Each video contained 100 frames, 320x180 pixels with 8-bit data. This work implemented the baseline visual pipeline approach using Matlab codes to implement the golden version for accuracy comparison. This work synthesized Verilog-HDL codes using Synopsis Design Compiler and VCS. The
Synopsis 28nm Generic Library is used for our power consumption results. The results are illustrated for the 4-channel Gabor Feature Extractor. The architecture is based on Figure 2.9. The error of the feature extractor and saliency modules are calculated using the below equations to obtain relative difference in pixel values. While not reported here, this work finds the small error rates also have insignificant impact on perceptual outcome of our object detection system. However, this work
uses this more abstract error metric to show generality of the approach beyond the perceptual results of our system.

\[ \text{Average}(|Out - Out'|) = \text{AverageDifferential}(1) \]

\[ \text{Maximum}(Out) - \text{Minimum}(Out) = \text{ValueRange}(2) \]

\[ \frac{\text{AverageDifferential}}{\text{ValueRange}} = \text{Errrorrate}(3) \]

Equation (3) describe the error calculation method for this section. Figures 2.12 and 2.13 show the averaged result across for four channels of a Gabor feature extractor for a sample video using a similarity threshold of 1. The feature extractor system starts with channel 1 and 3 employing just similarity filter and channel 2 and 4 exploiting channel redundancy in addition. Once a reference SI and SO are obtained, frame 2 can use temporospatial redundancy reducing the amount of computation due to the large increase in uniform regions observed in Figure 2.12. However, it induces very little error in the computation as observed in Figure 2.13. However, this work observes that, in frame 3, the computation rate increases by more than 5% compared to frame 2. Hence, this is an indicator the current frame has diverged from the reference frame. So, for frame 4, the SI and SO reference frames are refreshed and the computation modes of the channels mimic that for frame 4. The actual frame for which this mode change is triggered varies across
Figure 2.12: Reduction in computations due to exploiting redundancy for Feature extractor for a sample video.

Figure 2.13: Influence on Error Rate on the Gabor Feature extractor.

Figure 2.14 depicts the average computation rate when the threshold for the similarity filter is varied to provide additional savings. In the moving camera case, temporospatial redundancy is lower. Therefore, this system needs to update SI and SO frequently. However, in fixed camera case, background does not move. So, this system can have more chance to reuse SI and SO for many frames and reuse computations. The corresponding impact on error rate is shown in Figure 2.15. This work observes that even for a high threshold of 8, the error is less than 1%, indicating substantial potential for power savings. This work notes similar behavior for the salience module but omit the details for brevity.
Figure 2.14: Variation in computation reduction for different thresholds for the feature extractor averaged across all videos for moving and fixed cameras.

Figure 2.16 shows the power consumption of the Gabor Feature extractor employing 6 orientation and 6 scales (5x5, 7x7, 9x9, 13x13, 17x17, 19x19 windows) using Synopsis 32/28nm Generic Library [5]. The overall power consumed when employing different filters are shown. This work finds that the power consumption reduced by about 5 times when employing the exploitation of temporospatial redundancy and similarity filter of approach of this work for the various. These numbers average the savings across all 30 videos. The overheads such as the buffers, similarity filter and other additional logic are included in the evaluation and are observed to be small.
2.5 Related work

Approach of this work focuses primarily on dynamic power reduction, and uses clock and value gating techniques similar to those in used in [6,13,14,20,21], although with a very different policy. David Brooks [6] introduced dynamic adaption of data bit width to reduce computational power resource in CPU. Jason Cong [13] presented a Composable Heterogeneous Accelerator-Rich Microprocessor design that provides reconfigurable design for accelerator-rich structure to save power consumption. Louis [6] proposed a framework effectively implements data reuse through aggressive loop transformation-based program re-structuring. This work is for iterative loop in the programming that is software aspect. Vogelstein [6] provided an arbitrary network topologies between CPU, memory and digital-to-
Figure 2.16: Power Consumption for Gabor Feature Extractor.

analog converter for low power vision sensor. Sungho presented [21] a reconfigurable Network-on-Chip platform suitable for implementing domain-specific hardware accelerators in a design efficient manner. This reconfigurable platform switches the accelerator to optimize the vision tasks. These works had same or very similar goal with this work. This work applies gating even when the value does change, within a threshold, as the visual system is more error tolerant than their target domain of width reduction for data types smaller than the bus size.

Other power-saving proposals are more aggressive, and suggest supply voltage gating for visual systems and distributed approach [7,12,15–19]. Yoshimi Asada [7] introduced low power technology for image processing that used multiple power and power switch. Low speed mode used low power supply voltage to reduce...
dynamic power. And power switch disconnect the power line for disused part of hardware to reduce leakage power. Faro [16] presented Adaptive Background Modeling Integrated with luminosity sensors and occlusion processing for reliable vehicle detection. This work stored back ground image that has no interesting object in the scene. And it computed differential between input and back ground image. This work reduced computational resource using static stored back ground images only. Talla [18] analyzed multimedia workloads and proposes architectural enhancements for improving their performance on general-purpose processors. This work detected the instruction for parallel processing to save time resource. Jooyong [19] introduced real-time multi object recognition processor is presented with a three-stage pipelined architecture. This work used multiple CPUs for parallel processing for object recognition. Dominic [15,17] presented data driven process structure using subsystems. The parallel processing reduce running time and it also reduce the power consumption using data sharing. While this method cannot save leakage power, it also does not require long wake up times for the gated operator.

Window memoization has been proposed as an efficient method to accelerate visual processing. Farzad [8] presented Computational Redundancy in Image Processing using window memoization. This chapter focused on power saving, rather than speedup, and exploit runs of similar values in streaming image data to reduce transistor toggling by forcing similar inputs to a constant filtered proxy input. Thus, this technique does not require large memoization tables, as this work
only seeks to exploit local value consistency that can reduce power via eliminating transitions.

### 2.6 Sub Conclusion

This paper demonstrates three different methods to exploit naturally occurring forms of redundancy in visual information. Each of these three methods can be deployed orthogonally, and maximum savings are found when all three are deployed simultaneously. By relaxing this notion of sameness, this work expands the scope of the optimizations to decrease the fraction of inputs fully processed by up to 82% for a decrease in output accuracy of less than 1%. By synthesizing both the visual pipeline and the similarity filters, this work shows that the overhead required for detecting this approximate similarity is very modest in its area, power, and performance impact. Overall, the design improves power efficiency by 78% over the baseline design, even including the overheads for similarity filtering.
Chapter 3  |  Dynamic Prediction of Per-Input Saliency Algorithm Preference

Object classification, and other visual computing tasks, are increasingly important applications in both embedded and large-scale computing domains. Both the rapid pace of the field and the profoundly diverse system constraints of different embedded and analytic scenarios have together lead to a rich design space of algorithms for tasks such as selective search (SS) \cite{58}, objectness \cite{59,60}, and saliency \cite{36–42} that trade between accuracy and the cost (in either hardware complexity or execution time or both) of implementing/running the algorithms. To further complicate matters, while there is often a strong general correlation between implementation/execution cost and task accuracy across entire input sets, for a given input or input class, a cheaper approach will sometimes provide superior accuracy.
The input-dependent nature of the accuracy-efficiency tradeoff presents an opportunity to employ cheaper solutions on inputs where the added complexity of more robust solutions provides minimal or negative marginal value. However, to realize these benefits, a system would have to be able to accurately predict these opportunities, as the affinity will not be known a priori for a given input image. In this paper, goal of this work is to use a dynamic predictor that samples a video input to determine the preferred saliency algorithm within a visual pipeline. More concretely, many saliency detection methods [36–42] consider accuracy, in terms of Area Under the ROC Curve (AUC), and execution speed as the key metrics of interest, and approach of this work seeks to minimize execution time without negatively affecting AUC in the context of processing frames from videos.

To perform this prediction, this work trained a deep Convolutional Neural Network (CNN) [46] to select between two saliency algorithms with distinctive AUC/execution time tradeoffs [41,42]. This predictor forms the core of proposed dual saliency detection method and is based on a VGG-19 structure. The results for saliency detection show a slight improvement in AUC score with a substantial reduction in running time, despite a low reported predictor accuracy. This chapter provides insight into how a simple prediction accuracy metric is misleading for this use case, as many mispredictions are on cases of marginal benefit for one saliency approach over the other, whereas cases of high marginal benefit are strongly predicted. In addition to reporting computational resources and execution times,
this work presents detailed experiments about the sensitivity of this approach to the sampling based amortization in prediction overhead.

3.1 Related Work

3.1.1 Selective Search for Object Recognition

Selective search [58] was introduced by J. R. R. Uijlings in 2013. It generates possible object locations which combines exhaustive search such as sliding window and segmentation to obtain a small set of high-quality locations. The generated candidate object regions based on the scores can determine the amount of processing for the system resource.

3.1.2 Learning-based Salient object detection

Deep learning methods have achieved great performances for various purposes in computer vision, such as object detection, classification, and salient region detection. Rui Zhao [56] referred to saliency detection which uses the global-context and local-context. This structure has two branches for robustness and refinedness for saliency prediction. They have 16 convolutional layers and 3 sampling layers for two branches and one additional layer to merge these results. Five saliency detection data-sets are used for evaluation and this approach obtained the highest F-measure score among ten saliency detection methods.
3.1.3 Salient object detection ranking

Model of this work uses salient object detection ranking, similarly to Long Mai [56]. Multiple saliency detection algorithms generate saliency maps for each training image. These are evaluated for the performance using Area Under the ROC Curve (AUC score) as the popular quality measurement for saliency maps. The ranking based on pairwise preference is trained by three classification methods which are Random Forest Classifier (RFC) [52], Support Vector Machine (SVM) [47] and Multi-Layer Perceptron (MLP) [53]. These methods retrieve the best saliency map for a given image. 2,000 images were randomly selected for training and remaining 2,000 images for testing from THUS-10000 [57]. The experiments were repeated 10 times with random partition and reported the average results. This work differed the input of the predictor based on this method. It used multiple generated saliency maps. This work used only input images to predict the best saliency algorithm.

3.2 Motivation

Figure 3.4 shows a general example of object detection and classification structure. This structure has three parts which are feature extraction, ROI proposal and classification. Feature extraction algorithm [49–51], saliency detection and SVM were formerly popular combinations for this structure. Lately, Fast RCNN [48] is one of the popular structures which is trained for feature extraction and classification.
Figure 3.1: General structure for object detection and classification. ROI proposal provides location of object in the given image. Feature extraction optimizes information of image to analyze it faster in next step. Classifier discovers what ROI is.

using deep convolutional networks. These methods are evaluated using large test data-sets to present their performance for many cases of input scenes. To make an efficient system, the best algorithm should be used for each part. However, it is basically impossible to determine which is the best algorithm for a part based on its performance in isolation. The performance depends on the classification target, number of training data, background scene, input size, etc. In general, the method that scores highest in isolation is assumed to be robust in composition unless otherwise demonstrated. The chosen method has the highest probability to achieve better performance than others for the given scene.

Figure 3.2 shows saliency maps of six different algorithms. The performance can be changed according to the input scene and algorithm. If the system has multiple methods and analyzes the dependency for the input scene on run-time,
the system can choose the best method for all the parts to enhance efficiency.

3.3 Approach of this work

A key task in image analytics is to identify regions-of-interest, or ROI. ROI are areas in an image that are likely to contain objects of importance to the particular application. Once identified, ROI can be subsequently analyzed to detect the presence of these objects of importance. By focusing on object recognition based on ROI alone, high system throughput can be maintained without the loss of overall scene understanding functionality. Saliency-based ROI detection uses low-level visual features to generate candidate region proposals. While most saliency algorithms use low-level features, the difference in how these features are aggregated dictates the performance of the detector with respect to the quality of detected ROI. Additionally, depending on scene content different saliency algorithms will perform better or worse due to scene specific attributes such as object scale, presence of distracting textures, lighting variations, image noise, and image resolution. It is therefore advantageous to utilize an ensemble detector approach, in which the detector is dynamically selected from a bank of saliency detectors according to the scene content and structure. An investigation on using a trained Neural Network to provide the intelligence of this selection has been performed.

This proposed model consists of two saliency detections and trained predictor. The predictor retrieves which method can make a better saliency map for new
inputed images. To complete the trained model, VGG-19 [55] was used with pairwise comparison based on the AUC score. Prediction overhead for computational resources was also taken into account.

### 3.3.1 Saliency Map

This work chose two fast saliency detections among the methods which get average AUC score higher than 0.8 on the MIT Saliency Benchmark [43,44] two fast saliency methods were chosen. Fast and Efficient Saliency (FES) [41] and Spatially Weighted Dissimilarity Saliency (SWD) [42] which achieve 0.8 and 0.81 respectively in the evaluation with MIT Saliency Benchmark were selected. FES was introduced by Tavakoli in 2011. FES extracts salient regions of local features contrasted in a Bayesian framework on an average of 0.4 seconds. SWD was proposed by Duan in 2011. SWD took an average of 2.4 seconds to detect salient regions by sampling the patches from the input scene. Saliency maps were generated for all of the training and validation images in the ILSVRC-2015 object detection video (VID) dataset [45] which contains 361,822 training frames and 64,698 validation frames. The generated saliency maps were compared by the AUC score for every frame to identify labels. FES and SWD achieved an average of 0.667 and 0.76 for the AUC score for the training data-set. Although SWD generated a better saliency map for 75.4% of the training data-set, the system saved the computing resource when it used FES. To contemplate computational resources, a 10% advantage was
given to FES to choose FES when the performances were similar. 58% of training images prefer SWD over FES. All the tests for saliency detection were performed by means of MATLAB R2015b on a machine with Intel Core i7-5820k running at 3.3GHz clock, and 64GB RAM.

### 3.3.2 CNN Training

To predict the best method, CNN trained annotated data-set with Caffe library [54]. For efficient training, existing CNN structure and pretrained model are used in this work which is VGG-19. VGG-19 has 19 convolutional layers and three Fully-Connected layers. Input of VGG-19 is a fixed-size 224 x 224 RGB image. The number of final outputs of VGG-19 was modified to train data-set with two labels. The number of output channels in third Fully-Connected layer is changed from 1000 to 2. This CNN trains 361,822 training images with 64,698 images for validation from ILSVRC-2015 VID data-set. Each image has a binary label for two saliency detection methods. The pretraining weights were from training on the 1,000-class ILSVRC-2012 data set. This model extracts the compact feature from a given image for classification. CNN was initialized to the model for predictor except the third Fully-Connected layer. The learning rate was set to 0.0001 and initialized the fillers to a gaussian. On a system equipped with a NVIDIA GTX-980, training took about 4 hours for 30,000 iterations.
3.4 Experimental Evaluation

This section evaluates this method using the validation data-set in ILSVRC-2015 VID. The validation data-set has 281 videos that contain 64,698 frames. 109 videos that have more than 200 frames among the 281 videos were used for this evaluation. These videos contain 46,292 frames that was comprised of 54% of the SWD label and 46% of the FES label. The performance of saliency detection does not change frequently in the video. This work considered the sampling period to reduce overhead of prediction. This work demonstrated 1 to 300 sampling periods for this method. This work adopted the standard 10-crop testing [61] for the CNN input.

3.4.1 Evaluation

The average AUC score of SWD is 0.738 and FES is 0.6777 for validation data-set. When the predictor selects higher AUC score between SWD and FES for each image, the average AUC score is 0.7435. This number is the maximum score that can be achieved when the accuracy of the predictor is 100%. The accuracy of this predictor is 64.9% for all of the frames. The accuracy slightly decreases according to the increasing sampling period, because the decision of predictor is not changed frequently in the video. Two saliency detections with predictor achieved 0.7396 AUC score. This score is 0.0016 higher than the score of SWD and 0.0039 lower
than the maximum score. In figure 3.7, according to the increase of the sampling period to 300, the score decreases to 0.7341 that is 0.0039 lower than the score of SWD. This decrease gradually starts after around the sampling period 100.

Although the accuracy is 64.9%, this approach achieved a slightly higher AUC score than SWD. As the two AUC scores of saliency detections converges, the accuracy decreases. However, as the differential between two scores increase, the predictor selects the correct method. This is the reason why this work can achieve a higher AUC score than SWD with a low accurate predictor.

This work used a model of 30,000 training iterations to achieve this result. The accuracy increases with more training iterations, though the histogram of wrong selection case flatten. This flattening causes a lower average AUC score.

There are two options for CNN such as CPU mode and GPU mode. This predictor took 18 seconds on CPU mode and 0.27 seconds on GPU mode for CNN. In general, this predictor chose 64% of the test scene for SWD and another is for FES.

\[
\text{RunningTime} = SWD(2.33s) \times 64\% + FES(0.41s) \times 36\% + (Overhead/Samplingperiod)
\]

The running time with this work was 1.9 seconds when this work used GPU
mode with sampling period 1. This time is decreased to 1.65 seconds with sampling period 50. The running time is 19.65 seconds when this work used CPU mode with sampling period 1. This running time is decreased to 2 seconds with sampling period 50.

3.5 Sub Conclusion & Discussion

This paper proposes a dual saliency detection method with a predictor which selects the more efficient method, in terms of accuracy versus performance, among the given methods. This work employed a CNN as this predictor, training on 361,822 images in 30,000 iterations to classify the input scene based on an initial VGG-19 structure with a pretrained model. This work shows that the absolute prediction accuracy is misleading, as many mispredictions are on cases of marginal benefit for one saliency approach over the other. This is confirmed by the results for saliency detection actually showing a slight improvement in AUC score with a substantial reduction in running time. In addition to reporting computational resources, this chapter presents detailed experiments about the sensitivity of approach of this work to the sampling period that is used to amortize prediction costs. While the evaluation in this paper targets software, the same principles can be directly applied to hardware-accelerated saliency algorithms and predictors.
Figure 3.2: Saliency detection examples. Input (a) and (i) has same target object which is foreground region on Ground Trues (e) and (m). Another figures are results of these input by 6 different saliency detection algorithm. Although two input scene has same target object, these results are different by algorithm.
Figure 3.3: Examples to show performance based on weighted f-measurement. Images on top line are six sampled input frames in ILSVRC-2015 VID data-set. Second line is their ground trues. Third line and bottom line show thresholded saliency map from SWD and FES. The graph shows alterations of performance based on weighted f-measurement. These two lines are similar shapes. Performance of SWD is better than FES for most of frames. However, these performances are switched only between frame 74 and frame 101.

Figure 3.4: Dual saliency detection with predictor. Saliency predictor finds that which saliency detection method is better base on a given image. Saliency predictor provides enable signals and selection signals for two saliency detection methods and multiplexer.
Figure 3.5: Evaluation of FES and SWD based on AUC score. Blue dots indicate 60,000 of input images. Blue dots are on dashed line when the scores of two methods are same. Solid line which gives 10% advantage to FES is threshold to decide labels for training. Images which are relevant blue dots over the solid line are labeled for SWD. Others are labeled for FES.

Figure 3.6: Accuracy of prediction. The predictor selects correct method for 65% of test images.
Figure 3.7: Comparison of AUC Scores. Green line indicates AUC score of SWD (0.738). And Blue line is AUC score of FES (0.6777). These are straight lines to show their average AUC scores in the testing. Red line shows alteration of this method. This performance is slightly higher than SWD when it uses lower than 100 sampling period.

Figure 3.8: Computation time with prediction. Green line is the running time of SWD (2.33s). And blue line is the running time of FES (0.41s). These are straight lines to show their average running times in the testing. Red line is the running time when CNN is processed on CPU. Black line shows the running time when CNN is processed on GPU.
Within the domain of computer vision, object detection and classification is an active and ongoing field of research. With accelerated visual systems coming plausibly close to real-time processing within practical resource constraints, there is strong motivation for developing systems capable of both fast and effective object recognition in embedded scenarios, such as retail. For systems attempting to be more computationally efficient than exhaustive sliding-window searches, as is generally necessary in the embedded space, the most critical aspect of object recognition systems is the detection of the object(s) in question. This problem often becomes far more difficult in target-rich environments, such as a grocery store in which there are many items packed together, often irregularly, on the shelves. Further challenges may also arise from the quality of embedded video
inputs, although head-mounted wearable increasingly approximate actual visual fields.

One approach for selecting potential objects in a scene is to use visual attention models. Inspired by the human brain, such attentional algorithms attempt to identify the regions of an image which are most important or salient. There are several models for computing these regions derived by attention. One such algorithm, Attention using Information Maximization (AIM) [62–64] strives to achieve this through the use of statistical self-information contained with the image, identifying those regions which are most likely to provide the most information about the scene. Given both the resource constraints and the nature of inputs collected from embedded systems in the retail space, these approaches are attractive as a means of pruning the space of potential objects to classify. However, as this chapter will show in Section 4.3, visual saliency does not take advantage of domain knowledge about the likelihood of repeated object-types, which can be further exploited.

In this work, this chapter presents a multi-object detection and recognition system, Attention-Seeded Opportunistic Template-matching (ASOT), that involves a hybrid approach consisting of feature-based template matching as well as visual attention models to effectively locate and classify multiple objects within a visual scene. ASOT relies on saliency maps from AIM to generate initial object detection guesses. For each such salient region, ASOT performs template matching of
SURF [30] keypoints to perform object classification. ASOT also takes advantage of the likely repetition of objects to find additional objects that AIM may have missed. Once ASOT classifies a salient region, it then uses visual histograms to efficiently sweep the rest of the image for additional instances of the detected class.

The combination of attention-based object detection and exploitation of repetition allow ASOT to outperform prior work in recall and precision on the GroZi-120 data-set and provides opportunities for performance optimizations targeting embedded grocery and other retail environments.

The remainder of the paper proceeds as follows. Section 4.1 discusses related work. Section 4.2 outlines this proposed system, as well as the foundation on which it is based. Section 4.3 compares ASOT performance to prior work using the GroZi-120 [29] data-set and explores the impact of RoI scheduling extensions to ASOT that aim to maximize the number of objects recognized in scenarios where there are insufficient resources or time to investigate all salient regions. Section 4.4 concludes.
4.1 Related Work

This section discusses related work in attention-driven object detection and other object-recognition approaches targeting the grocery environment.

Visual attention algorithms are designed to identify the most interesting regions of an image. These regions are often found to contain the most conspicuous, informative, or interesting objects within a scene. There are a variety of ways in which these algorithms identify what pixels and regions are labelled as interesting. Visual saliency [65,66] attempts to model the human brain and the way it processes visual information. Such algorithms work to replicate the fixations of the eye as the image is understood. Other algorithms, such as AIM tackle the problem of region identification using information theory. These algorithms establish a probabilistic approach based on the self-information content of an image. Those regions found to contain the most self-information, and therefore the most information about the contents of the image are identified as the most important, or visually salient.

For object recognition in the grocery space, the works by George and Floerkemeier [31] and Merler et al. [29] are the most similar to proposal. George and Floerkemeier develop a system targeting grocery classification that divides the current image into grids at multiple scales, uses SIFT feature to apply hierarchical labels to each grid cell at each scale, and then combines results across scales using an image-level genetic algorithm to find plausibly consistent multi-label sets. That
is, the system is biased to select sets of labels that minimize the distance among labels across the externally provided label hierarchy. While they use SIFT, rather than SURF features, the template matching part of their technique is similar. The key differences with proposed recognition scheme stem from the gridded nature of their object detection in contrast with attention map model, which is not alignment constrained. Merler et al. introduce the GroZi-120 data-set and evaluate the performance of color histogram matching, SIFT template matching and boosted Haar feature matching. In contrast to this work, this work filter locations to test with template-matching using an attentional model, reducing total effort.

4.2 Detection and Classification using Attention-Seeded Opportunistic Template-matching

This section provides an overview of ASOT approach and discuss two extensions for exploiting multiple class instances per image.

4.2.1 ASOT-E(xhaustive)

The baseline version of ASOT, ASOT-E, involves a three-stage process for detection and identification of single object(s) within a scene. As a first step, the image is processed for extraction of SURF key-point features. Second, in addition to these key-points, this work also use a visual attention algorithm, Attention using
Figure 4.2: ASOT vision-processing pipeline, including RoI scheduling framework

Information Maximization (AIM) [64–66], to provide the likely locations of objects. The AIM algorithm produces a heatmap image $H$, indicating the relative salience or attentional importance of all pixels within the image. This heatmap is then converted into a binary image map by thresholding each pixel value such that any values below a given threshold are set to 0, while others are set to 1. For the experiments conducted in this paper, the threshold value was chosen to be the 70% percentile of the AIM-generated heatmap. Once thresholded, the binary saliency map is then processed using a connected component algorithm to identify contiguous regions of interest within the image. To filter out noise generated by the saliency algorithm, connected components are filtered out using several criterion, such as size and density within a particular region. Regions that are too
small cannot produce enough usable keypoints, while regions that are too large are often found to encompass too much of the total image, and not providing enough specificity and precision for useful detections. Regions that are found to cover a reasonable area of the image, but with limited density are also discarded for similar reasons. The resulting binary mask is then used as a filter and combined with the set of keypoints previously extracted from the image in order to identify salient regions of keypoints. Finally, it is these regions only, rather than the entire image, that are then matched against the template library to locate and identify objects in the scene.

4.2.1.1 Template Matching Phase

Each detected connected component region, \( R_q \), is assigned a saliency score, \( S_q \). This score is computed as the highest value found within the heatmap region that corresponds to the connected component region: 

\[
S_q = \max(H(R_q)B(R_q))
\]

The detected regions are processed sequentially, ordered by descending saliency score, through SURF template matching in an effort to find the best candidate match. The cost of performing exhaustive template matching is mitigated by first applying a stage of histogram comparison in order to select only those templates which are most likely candidates for the target region.

Histograms of visual words are generated for the target region. These histograms are then compared to similar histograms computed for each candidate template
using a Normalized Dot Product (NDP). During this comparison, only those visual words found to exist in both the image and the template are compared. The histogram comparison score for each template is taken as the sum of the NDP scores for the entire histogram of keypoints. These scores are sorted in descending order and only the top N highest matching templates are retained for matching in the next step. The number of templates retained must be chosen to strike a balance between recall accuracy and cost of subsequent template matching. In this work, only the top 5 highest scoring templates are selected for further matching against the target region. In order to optimize the matching of SURF features for each keypoint in the region, a KNN clustering algorithm is applied in order to identify the K most similar features within the template that could be a match, effectively pruning the number of keypoint comparisons that must be considered. In this work, K was chosen to be 32. The evaluation of keypoint descriptors is based on the result of the NDP between the region keypoint descriptor and each of the K candidate template descriptors. Using this metric, a perfect match will result in a value of 1.0 and so a suitable threshold must be chosen above which two keypoints are considered be a sufficiently good match. In this work, the threshold was chosen to be 0.98, requiring a 98% correlation between keypoint descriptors. For each region keypoint, the candidate keypoint with the highest viable match is selected to be the correct match within a particular template.

The template with the highest matching score is then selected as the winning
template, effectively classifying the detected object as the same class as the winning template. However, as the salient regions produced by AIM are not likely to encompass the entire object, additional work must be done in order to locate the bounding box of the target object. For each viable keypoint match, a two-dimensional affine transform is computed to relocate the template keypoint \((x_T, y_T, \theta_T, S_T)\) to the matched image keypoint \((x_M, y_M, \theta_M, S_M)\). This transform is computed assuming that the template is positioned on the image such that it is centered over the \((0,0)\) pixel of the image. The final transform is composed by averaging the transformations required for each matched keypoint. Using this transform, the bounding box which would cover the template, centered at \((0,0)\), is transformed to produce the target bounding box of the object within the image. This new bounding box is labeled with the matched template’s object label.

Figure 4.3: The left image shows pairs of matched features. A transformation matrix \(T\) is calculated by matching features with the best template image. The image on the right depicts the bounding box of the template translated and rotated into image space.
4.2.2 ASOT-O(pportunistic)

Having found a single object and its location in the image, this work posits, for the retail space, that it is very likely that there may be more instances of this object in the image. Using a template-matching-based search, this work attempts to find additional copies of this object. However, rather than using the existing template image, this work instead use the keypoints that lie within the bounding box generated for the newly detected object. Though this does not produce the clean keypoints that would typically be associated with a sanitized and ideal template, it has the added effect of compensating for any real-world conditions that may impact matching (such as lighting conditions, image quality, color adjustments, etc). Using this new region-template, each keypoint is then matched against all other keypoints in the image that are not contained within an already classified region. The matching of keypoints is evaluated in this phase is performed as before, selecting the maximum NDP score above the minimum threshold of 0.98 and again for each of these matches, a two-dimensional affine transformation is generated, this time with respect to the center of the image region. Using the center of the source region as the origin for the transformation allows for effective identification of likely candidate transforms using a 2D histogram derived from the values of the transformation matrices generated. The common coordinate system allows for all keypoint transformations related to a particular new instance of the object within a scene to be very similar, thereby increasing their histogram counts. Any
transformations which produce a sufficiently high histogram count are accepted and applied to simultaneously identify, locate, and classify likely copies of the source object. The thresholding of the histogram count in this work was chosen to be 2, indicating that if 2 or more keypoints produced the same transform, the resulting transformed region also likely contains another instance of the source object. Figure 4.4 shows an example of histogram filtering.

It is possible that some of the newly detected bounding boxes will cover or eliminate regions that were previously marked for processing by the AIM saliency detection algorithm. Due to this processing, those regions no longer need to be processed and template-matching begins again with the next remaining salient region. Processing for the entire image is complete when there are no more remaining salient regions to be processed.

Figure 4.4: The initial(left) figure shows the seed object(green) and matched features(red). These matches are described by a 2D histogram(middle) of \( tr^x \) and \( tr^y \) for matched features. The histograms are then filtered for noise(right). To reduce noise, this work used a Gaussian filter and thresholds.
4.2.3 ASOT-S(cheduled)

While the processing required in the template matching phase is always proportional to the number of templates, the processing required in ASOT-O to sweep the rest of the image for additional instances once a classification has been made is related to the number of unlabeled keypoints that remain in the image – essentially, the portion of the image that has not yet been labeled. Due to the fact that this phase is capable of detecting multiple copies of an object over a single iteration of the image, it follows that the overall processing could be reduced by finding and classifying those objects that are most prevalent first. Using these concepts, this work would like to schedule salient regions for processing based on combination of the likelihood of the corresponding prevalent of object within a scene as well as that importance of object to the scene.

As a first step, all keypoints in the image are matched using NDP against keypoints in each template. The score for each template, $T_q$ is computed by computing a value, similar to a probability likelihood, using the sum of the NDP scores for each keypoint in the template: $T_q = (1 - \frac{1}{1+\Sigma(A'_q)})$. A final score for each detected region is calculated the weighted sum of the scores: $Score_q = w_0 \times S_q + w_1 \times T_q$. The detected regions are processed in descending order of their computed score. Using this step to schedule the order in which the detected regions are processed, the image is then processed identically to ASOT-O.
Table 4.1: Performance of saliency map algorithms on Dataset for this work.

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Recall</td>
<td>0.66</td>
<td>0.52</td>
<td>0.25</td>
<td>0.61</td>
<td>0.33</td>
<td>0.58</td>
</tr>
<tr>
<td>Precision</td>
<td>0.64</td>
<td>0.62</td>
<td>0.7</td>
<td>0.65</td>
<td>0.66</td>
<td>0.67</td>
</tr>
<tr>
<td>F-measure</td>
<td>0.64</td>
<td>0.56</td>
<td>0.36</td>
<td>0.62</td>
<td>0.44</td>
<td>0.61</td>
</tr>
</tbody>
</table>

4.3 Experimental Results

For evaluation of these detection and classification systems, this work have examined the performance with respect to two data-sets. The first is a grocery dataset that was gathered and assembled in-house and the other is the more widely available GroZi-120 dataset [29]. The performance metrics this work have used in evaluation are mean average precision (MAP) and mean average product recall (MAPR), consistent with the work by George and Floerkemeier [31]. The processing was performed using MATLAB running on a 3.5GHz Intel Core i5 CPU with 16GB of system memory. For the experiments involving models of this work, the histogram comparison was limited to accepting only the top 5 leading candidates. An object is considered to be correctly detected if the region generated covers at least 70% of the ground truth region, and is considered to be correctly labeled if the matching template is of the same class as the ground truth label (i.e. both are laundry detergent, soda, etc).
Table 4.2: Performance on Grozi-120.

<table>
<thead>
<tr>
<th>Recognizing groceries [31]</th>
<th>mAPR</th>
<th>mAP</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASOT-S</td>
<td>15.54%</td>
<td>28.42%</td>
</tr>
</tbody>
</table>

4.3.1 Evaluation on the GroZi-120 dataset

To show the performance of this approach, this work compared best-performing model (ASOT-S) against the work by George and Floerkemeier [31] from ECCV 2014 using the GroZi-120 [29] data-set. This data-set consists of 120 classes of objects, taken both as images from the web as well as from frames from video shot in a real-world grocery store. For training purposes (template selection) the inVitro subset was used which consists of 2-14 images per class, with 5-6 on average. The inSitu subset, consisting of 885 low resolution images, averaging 7-8 per class, was divided and used to generate histograms for template filtering, as well as for evaluation (detection and classification). Due to the low resolution of the training set images, the histogram filtering performance optimization was ineffective, and was disabled for the GroZi data-set. Due to the intra-class variation in image counts in these data-sets, each class is represented using anywhere from 6 to 25 template images. The class histograms where generated using an average of 11 images per class. Their approach was found to achieve a mAPR of 9.37% and a mAP of 13.21%. Due to the structure of the images in this data-set, ASOT-S, -O and -E perform similarly, as the majority of the images contain only a single instance of each of
the products present in the image. However, using ASOT-S provides an mAPR of 15.54% and an mAP of 28.42% while also providing a performance boost over the other ASOT models. Despite the cost of additional preprocessing necessary to generate the improved ROI scheduling, as the number of candidate regions increases in an image, this cost is found to pale in comparison to the cost of the additional template matches required in order to cover all of the regions.

4.3.2 Impact of class repetition on performance

Table 4.3: ASOT performance on supermarket dataset.

<table>
<thead>
<tr>
<th>Model</th>
<th>Recall</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASOT-E</td>
<td>32%</td>
<td>80%</td>
</tr>
<tr>
<td>ASOT-O</td>
<td>50%</td>
<td>81%</td>
</tr>
<tr>
<td>ASOT-S</td>
<td>47%</td>
<td>81%</td>
</tr>
</tbody>
</table>

Table 4.4: Time to reach specified fraction of recall saturation.

<table>
<thead>
<tr>
<th>Model</th>
<th>0.6x</th>
<th>0.7x</th>
<th>0.8x</th>
<th>1.0x</th>
</tr>
</thead>
<tbody>
<tr>
<td>ASOT-O</td>
<td>7.23s</td>
<td>8.01s</td>
<td>8.48s</td>
<td>9.26s</td>
</tr>
<tr>
<td>ASOT-S</td>
<td>6.67s</td>
<td>7.71s</td>
<td>8.61s</td>
<td>9.76s</td>
</tr>
</tbody>
</table>

Table 4.5: Performance of ASOT-E on the Dataset.

While the Grozi-120 data-set features limited repetition within an image (the source material is from a convenience store that stocks few of any one item), supermarket-style grocery stores can feature high instance counts from each class.
To explore the impact of this high-repetition environment on ASOT, I developed in-house grocery data set for this work. In-house grocery data-set of this work was assembled using 175 images taken in real-world grocery stores using a simple mobile phone. These images contain approximately 3072 individual instances of identifiable 49 product templates that this work has grouped into 17 classes such as laundry detergent, cookies, cracker and cereal. Examples of some of the product templates and classes can be found in Table 4.6 and example retrieval output is shown in Figure 4.7.

Figure 4.5 shows the three ASOT variants running on an sample image from data-set for this work. Table 4.3 shows the performance of each ASOT model on in-house 175 test-image data-set and Table 4.4 compares the rate at which ASOT-O and ASOT-S reach their recall saturation point. For all experiments on this data-set, this work chose the top five candidates in the histogram comparison step. ASOT-E, which does not exploit retail-specific assumptions, produces a recall of 32% and precision of 80%. Note that higher overall responses than GroZi-120 are expected due to the reduction from 120 to 17 classes. ASOT-O, exploiting the assumption of repetition, achieves a recall of 50% and precision of 81%. ASOT-E suffers a slight reduction in recall to 47% while maintaining a precision of 81%. The alteration of scheduler order with ASOT-S improves run-time despite the initial preprocessing overhead up until 0.7x of the recall saturation point is reached, after which ASOT-O catches up, as most high-value matches have been found.
If early termination was necessary due to resource constraints in an embedded scenario, ASOT-S may outperform ASOT-O in practice. However, it also appears to over-eagerly prune salient regions that overlap with detected matches from the current class, reducing recall from 50% to 47%. Additional tuning of the conditions for removal from the processing list may mitigate this negative impact.

<table>
<thead>
<tr>
<th>Class 1</th>
<th>Class 2</th>
<th>Class 3</th>
<th>Class 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><img src="image1.png" alt="Image" /></td>
<td><img src="image2.png" alt="Image" /></td>
<td><img src="image3.png" alt="Image" /></td>
<td><img src="image4.png" alt="Image" /></td>
</tr>
</tbody>
</table>

Table 4.6: Multi-template classes in in-house grocery dataset

4.4 Discussion

This work has proposed two new models for the detection and classification of objects based on cues derived from visual attention algorithms. In addition to these attentional cues, these models exploit repetitive visual information to enhance detection and classification, particularly within a target-rich environment such as a grocery store or supermarket. This chapter has shown that intelligent scheduling of the processing for such attention-directed regions can further improve the performance of such a system. These models have been evaluated using two grocery-themed data-sets derived using real-world visual imagery. These results demonstrate that by reusing information found in the scene, alongside attention
and intelligent scheduling of region processing this chapter can significantly improve not only the detection (recall) of objects, but also their classification accuracy (precision) as well. Further improvements to the initial attention-based region detector, such as the use of a hybrid ROI detector are being explored in order to increase the effectiveness of these proposed systems.
Figure 4.5: Output and recall/time performance of ASOT-E, -O, and -S on example image
Figure 4.6: ASOT Retrieval Examples (1)
Figure 4.7: ASOT Retrieval Examples (2)
Chapter 5

Conclusion

This dissertation presented three different approaches to make an efficient vision system. These approaches included exploiting redundancy in visual information to reduce power consumption, dynamic algorithm selecting to optimize vision system on run-time, and task scheduling to produce better quality on the vision system which has limited resource. Three types of redundant information across pixels, channels and frames were exploited to save computational resource. Spatial similarity was used to reduce redundancy in a given image. The computation was skipped and reused the previous result when the current input and previous input was similar. To reduce redundancy across the channels, several filters compared the inputs across the channels such as RGB, HSI and YCbCr. The neuro-inspired vision pipeline that is our case study has four input channels.

These approaches reduced 78% of the redundant computation in the vision pipeline. The linear computation which was convolution in this work was used for
a lot of vision applications. For instance CNN which is the most popular machine learning algorithm these days consists with many convolutional layers with multiple channels. CNN has proliferated to many areas such as voice recognition, economic analysis and security. CNN needs a large number of iterations with a million of training data-sets. This work can reduce power consumption for both training and application. A single algorithm cannot efficiently perform for all situations such as indoors, outdoors, dim lights, complex scenes and blurred scenes. An algorithm has pros and cons. However, it is tough to find which algorithm works well for which scene. Machine learning was the appropriate method to compare the performances. CNN was selected to make the learning model with image-net VID data-sets. Two algorithms which are SWD and FES were compared to measure their performances to make annotation. The AUC score was the first target and the running time was second target. This approach achieves a slightly higher AUC score than SWD. And the average running time was faster than SWD. Most of the researchers focus on developing better methods. Their work is valuable for the future. However, multiple existing methods can be collaborated for better performance. This work presented how to collaborate multiple methods. This approach needs to go hand in hand with developing new methods to maximize the performance.

To provide better (better) quality of object detection on mobile systems, the task scheduler set the priority for the scenes that has multiple and duplicated objects. The difficulty of the ROI proposal for grocery scenes was number of
objects. A general ROI proposal could not create bounding boxes for many objects. The interesting fact about grocery scenes was that they have a lot of reusable information from duplicated objects. A system which has limited execution time should be able to detect an object that can be reusable in order to detect more objects in the given time. The task scheduler found duplicated ROIs in the scenes and counted it. Template matching was used on salient regions to make object bounding boxes. SURF matching found similar ROIs in the scenes. The best case of object detection is that it allows the system to have enough computational and time resource. In general, a highly accurate method requires more complex computation and longer running time. The ordering of priority is an efficient approach for the mobile devices that have limited resources to decrease the unit price. The system can perform more tasks in given time. This dissertation contributed to the power efficient vision hardware design exploiting three redundancies for battery life which is one of the most critical problems on mobile vision systems. Algorithm choice for the application is the biggest issue in design time. Many candidates should be considered and a lot of data should be tested to choose the best method for the application. Dynamic algorithm selection based on the predictor and multiple algorithms optimized the vision system on run-time. The algorithm choice issue can be eased off by multiple algorithm selection.
Bibliography


Vita
Chris S. Lee

Chris Songjin Lee is a PhD candidate in the Department of Computer Science and Engineering at the Pennsylvania State University since 2012. His primary research interests are in hardware accelerators, embedded, system-on-chip and computer architecture, hardware software co-design and neuromorphic systems.

Education

Pennsylvania State University University Park, Pennsylvania 2012.1-2016.12
Ph.D. in Computer Science and Engineering
Advised by Prof. Vijaykrishnan Narayanan

Dong-A University Busan, South Korea 2007.3-2009.2
M.S. in Electronics Engineering
Advised by Prof. Bongsoo Kang

Dong-A University Busan, South Korea 2000.3-2007.2
B.S. in Electrical Engineering

Selected Publications


