The Pennsylvania State University
The Graduate School
College of Engineering

QUADRATIC BINARY OPTIMIZATION
FOR PEDESTRIAN DETECTION IN CROWDED SCENES

A Dissertation in
Computer Science and Engineering
by
Sitapa Rujikietgumjorn

© 2014 Sitapa Rujikietgumjorn

Submitted in Partial Fulfillment
of the Requirements
for the Degree of

Doctor of Philosophy

August 2014
The dissertation of Sitapa Rujikietgumjorn was reviewed and approved* by the following:

Robert T. Collins  
Associate Professor of Computer Science and Engineering  
Dissertation Advisor, Chair of Committee

Kamesh Madduri  
Assistant Professor of Computer Science and Engineering

Jesse Barlow  
Professor of Computer Science and Engineering

Paul Griffin  
Department Head and Professor of Industrial and Manufacturing Engineering

Lee Coraor  
Associate Professor of Computer Science and Engineering  
Director of Academic Affairs

*Signatures are on file in the Graduate School.
Abstract

Pedestrian detection is of much interest in the computer vision research community and is a rapidly evolving research area. Significant progress has been demonstrated but the performance degrades rapidly when occlusion occurs. We propose a binary quadratic optimization framework for multi-pedestrian detections to improve pedestrian detection, especially when there is an occlusion. The pedestrian detection problem is formulated as a Quadratic Unconstrained Binary Optimization (QUBO) problem. The formulation reasons directly over the space of overlapping among object detections as a pairwise measurement and an individual detection confidence score as a unary measurement. The tradeoff between detection confidences and amounts of overlap is optimized thereby allowing a more relaxed selection compared to conventional non-maximum suppression approach. Although QUBO is an NP-hard problem, efficient approximate methods are available, and these yield high quality solutions on large problem sizes.

The core concept of optimized detection framework is to generate a large set of possible detection candidates and then use the optimization method to select a subset of those candidates that best represents the detection. The optimized detection framework is further applied to a multiple-body-part representation called “body plan”. With the adoption of multiple parts, reasoning about occlusions among pairs of overlapped detections, by applying a positive (reward) and negative (penalty) pairwise scores, can help to decide whether the missing body part is likely due to being occluded. Besides detection candidates, it is also applicable to use tracklets of matching as candidates and formulate the problem as non bipartite matching graph. We reformulate the non bipartite matching problem into a Quadratic Constrained Binary Optimization that solves for a set of detections along with data association. The proposed optimized detection framework shows that quadratic optimization for reasoning about overlapping detections and quality of an individual detection can improve the performance of a pedestrian
detection system, especially when there are multiple occluded persons. We also show that the framework of optimized detection can be applied to several types of candidates, not only a sliding window based detector but also shape covering and pairs of matching candidates.
# Table of Contents

List of Figures viii

List of Tables xii

Acknowledgments xiii

Chapter 1
Introduction 1
1.1 Pedestrian Detection ............................... 3
1.2 Overview of Optimized Detection Framework ......... 7
    1.2.1 Candidate Detection .......................... 8
    1.2.2 Optimization ................................ 9
    1.2.3 Extension as a Data Association Framework ... 11
1.3 Thesis Outline ................................. 12

Chapter 2
Related Work and Background 13
2.1 Related Work ................................... 13
2.2 Quadratic Unconstrained Binary Optimization (QUBO) 20
2.3 Solving QUBO Problems ........................... 21
    2.3.1 Greedy Forward Search ........................ 22
    2.3.2 Tabu search ................................ 24
    2.3.3 Quadratic Programming ....................... 25
    2.3.4 Branch and Bound ............................. 27

Chapter 3
Optimized Pedestrian Detection for Multiple and Occluded People 32
3.1 Introduction .................................................. 32
3.2 Optimized Detection Framework ........................... 34
  3.2.1 Generating Dense Detection Candidates for Pedestrian Detection ........................................... 35
  3.2.2 Formulation as a QUBO Problem ................................. 41
3.3 Experimental Results ........................................... 46
  3.3.1 Evaluation Datasets ........................................... 46
  3.3.2 Results ...................................................... 47
3.4 Conclusion ...................................................... 53

Chapter 4
Improving Occluded Pedestrian Detection by Modeling Unoccluded Body Parts 59
4.1 Introduction .................................................... 59
4.2 Related Work ................................................... 61
4.3 Detection Candidates .......................................... 62
  4.3.1 Generating Multiple Body Parts ............................ 63
  4.3.2 Generating Body Plans by Combining Parts ................. 65
4.4 Our Approach .................................................. 67
  4.4.1 Overlap Among Candidates .................................... 68
  4.4.2 Occlusion Reasoning ......................................... 69
  4.4.3 Problem Formulation ......................................... 72
  4.4.4 Weight Parameter Learning ................................... 74
4.5 Experimental Results ......................................... 75
  4.5.1 Results ...................................................... 76
  4.5.2 Quality of the Solution ....................................... 80
4.6 Conclusion ...................................................... 81

Chapter 5
Optimized Detection Coupled with Data Association 82
5.1 Introduction .................................................... 82
5.2 Our Approach ................................................... 84
  5.2.1 Generating Tracklet Candidates ............................. 87
  5.2.2 Problem Formulation ......................................... 90
  5.2.3 Non-Bipartite Graph Reformulation ......................... 91
5.3 Experimental Results ......................................... 95
5.4 Summary ....................................................... 96
Chapter 6

Conclusion and Future Work

6.1 Conclusion ................................................. 99
6.2 Related Publications ..................................... 102
6.3 Future Works .............................................. 102

Bibliography ................................................. 104
## List of Figures

1.1 Examples of (a) low density, (b) medium density, and (c) high density pedestrian scenes. ........................................... 2

1.2 Example images from INRIA datasets illustrating variability of pose and appearance, as well as different types of occlusions. ............. 3

1.3 (a) PLS detector with non maximum suppression misses one person. (b) Candidate object center locations from the PLS detector without non maximum suppression show that there were correct candidates there, but they were suppressed by stronger neighboring detections. ........................................... 5

1.4 (a) HOG-based human detector from OpenCV run at one scale level. (b) The detector confidence map at that scale level prior to non maximum suppression. ........................................... 5

1.5 Overview of proposed framework using full body detector (left), and multi-part detector (right) to generate an initial set of candidates. .... 8

1.6 Concept of framework extension to tracking. (a) Matching candidates (b) Optimized solution with matching shown in the same color. ........................................... 10

2.1 An example illustrates steps for a branch and bound to further divide a subproblem or fathom a node, depicted as a red cross. ........ 30

3.1 An example of optimized detection ........................................... 33

3.2 Overview of our proposed optimized detection framework. See text for details. ........................................... 36

3.3 An example of computing unary score from foreground pixels using an elliptical shape mask. The score for this example is 2307. .... 37

3.4 A set of elliptical shape candidates in (b) is generated from a foreground image in (a) by the shape covering approach. ............. 38

3.5 A multi-scale confidence score map at four different scales. ........ 39
3.6 An example set of candidates generated by detector confidence map sampling. (a) Confidence score map from a single scale level of the multi-scale detector confidence maps. (b) Candidate samples generated from it by random sampling. ........................................... 40

3.7 Expected human sizes learned as a regression function between y coordinate location in the image and expected height of a detected pedestrian at that location. ................................................................. 42

3.8 Using quadratic binary optimization to find the best shape covering of a foreground mask. .......................................................... 44

3.9 Example images from three evaluation datasets. (Left) dataset A: CVLab’s Terrace sequence dataset; (right) dataset B: PETS 2009 dataset. ................................................. 47

3.10 A result from each step of our proposed approach using two different detection approaches for generating candidates. The top row uses shape covering to produce a dense set of candidate detections. The bottom row uses a sliding window human detector. (a) Foreground image. (e) Confidence map from a human detector. (b)(f) Candidate detections exceeding a confidence threshold. (c)(d)(g)(h) Final set of optimized detections shown in yellow ellipses. False negatives are shown in red. ............................................................... 48

3.11 ROC curve comparison between Tabu search, Greedy, and Quadratic programming using a sliding window human detector to generate initial candidates. ................................................................. 49

3.12 Evaluation result of 5 methods (our 3 approaches, OpenCV’s HOG-based detector and PLS detector) on dataset A. ......................... 50

3.13 Evaluation result of 5 methods (our 3 approaches, OpenCV’s HOG-based detector and PLS detector) on dataset B. ......................... 51

3.14 Comparison result of our approach with other methods on dataset A 54

3.15 Comparison result of our approach with other methods on dataset B 56

4.1 Example cases where a person is occluded but some body parts are clearly visible. (a) visible head, torso, and leg (b) visible head and torso (c) only head is visible. .................................................. 60

4.2 An example of the person model. (left) Several higher resolution part filters. (middle) Spatial model for the location of each part. (right) The filters specify weights for histogram of oriented gradients features .................................................................................. 63

4.3 Grouping seven part filters of the Deformable Part Model detector [19] into head (left), torso (middle), and legs (right). ...................... 64
4.4  (a) head detection candidates (b) torso detection candidates (c) leg 
detection candidates (d) all part detection candidates from (a)-(c) . 65
4.5  Examples of different types of body plans (a) head (b) head-torso 
(c) head-torso-leg . . . . . . . . . . . . . . . . . . . . . . . . . . . . 66
4.6  Overview of our proposed multi-part body plan detection framework. 67
4.7  Two examples of a pixel overlap area between two body plans. Left: 
overlap between a two-part and three-part body plan. Right: over- 
lap between two three-part body plans. . . . . . . . . . . . . . . . . 69
4.8  Occlusion reward ratio between a body plan hypothesis (red, green, 
and blue boxes) and a neighboring body plan hypothesis. Solid lines 
represent detected parts and dotted lines represent missing parts. 
For Person A, (b) and (d) are better candidates than (a) and (c) 
respectively, as explained in the text. . . . . . . . . . . . . . . . . . 71
4.9  Matrix $Q$ for objective function in eq. 4.2 . . . . . . . . . . . . . 74
4.10 Graph showing cost function (1-accuracy) during each iteration of 
the pattern search procedure that solve for relative weights $w_1, w_2, w_3, w_4$
that maximize system detection accuracy on a training set. . . . . . 75
4.11 Qualitative evaluation shown as ROC curve. (top) Terrace Dataset 
(middle) Caviar Dataset (bottom) Art Fest Dataset. . . . . . . . . . 77
4.12 Our results are shown in (a) and (b) where (a) shows detected body 
parts and (b) shows final bounding boxes. (c) Optimized detection 
(Chapter 3) (d) Grammar model [25] . . . . . . . . . . . . . . . . . . 78
4.13 Examples of some incorrect detection results using our method such 
as placing a body plan on an incorrect part of the body or a head 
floating near another candidate. . . . . . . . . . . . . . . . . . . . . 79
4.14 Actual solution values solved by quadratic programming on the 
relaxed continuous problem. In all cases examined, the continuous 
solution vectors were indistinguishable from a binary solution. . . 80

5.1  Example frames showing an unoccluded person in the first frame 
being occluded in the next two frames. . . . . . . . . . . . . . . . . 83
5.2  (left) A set of detection candidates shown as colored bounding 
boxes. (right) Graphical representation of the set of detections in 
the left image. The color of the node indicates which candidate in 
the left it belongs to. . . . . . . . . . . . . . . . . . . . . . . . . . . 85
5.3  A graphical representation of candidate body plans generated in 
two frames. . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . 86
5.4  A feasible candidate set of tracklets, represented as blue edges. Se- 
lecting a subset of these edges also selects a subset of x and y nodes. 87
5.5 Adding dummy nodes, $X_0$ and $Y_0$, for a selected candidate with no-match.

5.6 An example of to non-bipartite matching graph reformulation. (left) A non-bipartite matching graph with three candidates, $X_1$, $X_2$, and $X_3$, in the first frame and two candidates in the second frame, $Y_1$ and $Y_2$. Its linear constraints are shown on the bottom left. (right) The reformulated graph is represented as a quadratic constrained binary optimization problem. The square nodes represent undirected edges in the left graphical model. Color edges depict the regenerated pairwise edges.

5.7 (a) Frame 455. (b) Frame 460. The subsequent frame of (a) and (b) are shown on the right. The numbers on bounding boxes indicate matching id. Detection bounding boxes are shown as full body (yellow) and body parts (red, green, and blue).

5.8 (a) Frame 480. (b) Frame 505. The subsequent frame of (a) and (b) are shown on the right. The numbers on bounding boxes indicate matching id. Detection bounding boxes are shown as full body (yellow) and body parts (red, green, and blue).

6.1 Overview of the optimized detection concept used in three proposed frameworks.
List of Tables

3.1 Quantitative comparison of the 5 methods on dataset A. 52
3.2 Quantitative comparison of the 5 methods on dataset B. 52

4.1 Comparison of true positive, false negative, and false positive percentages. 76
4.2 Comparison of accuracy and precision percentages. 76

5.1 Quantitative result for simultaneous detection and matching across pairs of frames. Table shows percentages of true positive (correct), false negative (miss), false positive, accuracy, and precision. 96
Acknowledgments

Without the love, support, guidance, and expertise of many wonderful people, my Ph.D. study would not have been accomplished. Hereafter, I would like to take this opportunity to acknowledge all of these people. First and foremost, I would like to express the deepest appreciation to my advisor Assoc. Prof. Robert T. Collins for being a great mentor and giving me an opportunity to learn and work in this field. His guidance, substantial advice, belief in my abilities, and extensive knowledge provided invaluable support and inspiration to me.

In addition, a grateful thank to my committee members; Asst. Prof. Kamesh Madduri, Prof. Jesse Barlow, and Prof. Paul Griffin, for their insightful opinions and comments. I am thankful to Prof. Yanxi Liu for serving as my former committee member and for giving me valuable comments. I sincerely want to thank LPAC members; Jingchen Liu, Asad Butt, Christopher Funk, Jesse Scott, and Ingrid Rauschert, for providing excellent commentary on my work and the support I needed. I also want to thank all my Thai friends at Penn State for the wonderful memories. I would also like to express my appreciation to the support from the Royal Thai Government, the Penn State Department of Computer Science and Engineering and the National Science Foundation.

Lastly and most importantly, I would like to thank my mom and dad, Suttirat and Somkiet Rujikietgumjorn, for all their love, support and encouragement. Mom, I always remember the Thai proverb you always said to me “forgo the sour for the sweet”. To P’Ming, who always be there to help me, I am glad to have you as my big brother. To Nhui, I want to thank you for being a part of my life and helping me getting through my difficult times. You made me realized how truly lucky I am to have you in my life. I would like to dedicate this work to them. Without them, I would not have reach this milestone in my life.
To my mom, my dad, P’Ming, and Nhui.
Chapter 1

Introduction

Digital video cameras are now widely available and becoming more popular for use in applications such as surveillance, traffic analysis, and event monitoring. Instead of having a human operator doing all the work, it is desirable to process the video automatically to gain information as needed, for example, to track people and analyze their walking paths, activities, or interactions. Because of the demand for automated analysis, research in computer vision has gained more interest in several areas such as medical analysis, robotics, and photogrammetry. One of the main research topics in computer vision is object detection, since the ability to detect and track objects is useful for nearly all video applications. In this thesis work, we focus on detecting pedestrians. The goal of pedestrian detection is to locate where a person is in an image such that location of people can be tracked over the following consecutive frames.

Pedestrian detection becomes more difficult when there are multiple people in the same scene, especially for a group of people or crowd where occlusion is commonly occurring. Figure 1.1 shows several pedestrian scenes with different density levels. In most pedestrian dynamics studies, “density” is defined as $D = \frac{N}{A}$.
where \( N \) is a number of pedestrians and \( A \) is an area of size \( A \) \cite{68}. Detecting a person in a single-person scene is easier to accomplish than detecting a person in a group, but many real world environments contain more than one person. A pedestrian detector that does well in a single-person scene may have a large decrease in performance when applied to a scene with more people, particularly in a scene that contain occlusions. A high density crowd is even more difficult to analyze since most body parts are occluded and individual people have low resolution because crowd scenes are usually recorded with a wide field of view.

Our goal is to be able to detect all persons in a scene up to a medium level of density, including those who may be partially occluded. Occlusions may occur slightly, partially, or even totally. Handling total occlusion requires information

\footnote{Images from \url{http://pascal.inrialpes.fr/data/human/} \newline \url{http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/} \newline \url{http://www.cvg.rdg.ac.uk/PETS2009/a.html} \newline \url{http://crcv.ucf.edu/data/crowd_counting.php}}

Figure 1.1: Examples of (a) low density, (b) medium density, and (c) high density pedestrian scenes. \footnote{Images from \url{http://pascal.inrialpes.fr/data/human/} \newline \url{http://groups.inf.ed.ac.uk/vision/CAVIAR/CAVIARDATA1/} \newline \url{http://www.cvg.rdg.ac.uk/PETS2009/a.html} \newline \url{http://crcv.ucf.edu/data/crowd_counting.php}}
beyond a single image frame such as the previous location and velocity of a person as determined by tracking, and is beyond our current goal. In this work, we want to able to detect partially occluded persons who still have some visible parts in the image, such as head or upper torso.

1.1 Pedestrian Detection

Detecting people is a difficult problem due to body pose articulation and variation in human shapes and appearances. Arms, legs, and articulated joints allow people much more flexibility than rigid objects such as cars, making people more difficult to detect. Clothing with similar color to the background also makes it difficult to detect. Figure 1.2 shows examples of pedestrian from INRIA datasets illustrating some difficulties such as a difficult pose in dancing or sitting, a person occlusion, and an object occlusion.

Figure 1.2: Example images from INRIA datasets illustrating variability of pose and appearance, as well as different types of occlusions.\textsuperscript{2}

For pedestrian detection, there are several types of partial occlusion that can occur as shown in Figure 1.2. The first type is self occlusion, which occurs when hands, arms, or legs are hidden behind other body parts like the torso. Another type is people occlusion, where a person is partially hidden behind another person.

\textsuperscript{2}Images from http://pascal.inrialpes.fr/data/human/
in a scene with multiple people. This issue becomes more problematic when there are many people at different scales and locations in the scene, which is common in surveillance video. The last type of occlusion is an object occlusion, which is a person being occluded by an object in the scene such as a table, a chair, or a shrub.

Occlusion makes detection harder because some parts of the body are not seen. That usually results in a weak or missed detection when a detector has been trained using the whole body. A detection response of an occluded person usually has low confidence score and is likely to be suppressed. The difficulty in handling low confidence detections in an occlusion scene is deciding whether a detection response from a pedestrian detector should be discarded as a false detection or should be kept because it may belong to an occluded person. Either decision could be wrong, leading to a false negative or false positive detection.

Conventional detection methods use a sliding window based approach to apply the detection at multiple locations and scales in the image. However, this approach commonly leads to multiple responses of the detector on the same individual at slightly shifted spatial locations and neighboring scales. To prune these multiple responses, non maximum suppression (NMS) is applied to remove low-confidence detections that overlap a higher confidence one. However, in the case of two or more people who are close together or even partially overlapping, non maximum suppression may suppress a correct detection, resulting in a missed detection. Figure 1.3 shows a result from using the Partial Least Square (PLS) detector from [64] with its default non maximum suppression routine enabled. In this example, the detector misses one person; however, a closer inspection of detection candidates prior to non maximum suppression reveals that there were candidates generated at the correct location, but they were suppressed. Another example in Figure 1.4
Figure 1.3: (a) PLS detector with non maximum suppression misses one person. (b) Candidate object center locations from the PLS detector without non maximum suppression show that there were correct candidates there, but they were suppressed by stronger neighboring detections.

Figure 1.4: (a) HOG-based human detector from OpenCV run at one scale level. (b) The detector confidence map at that scale level prior to non maximum suppression.

shows a result from OpenCV’s HOG-based human detector. One person in the group of five people is not detected. However, the detector confidence map at that scale shows high responses at the missed location that were suppressed by stronger neighboring detections.

Our motivation is that we want a better and smarter way to select or discard a detection response from pedestrian detectors. We want a detection framework that reasons directly over the space of overlap or occlusion when selecting a detection.
response. The detection framework should allow a neighboring detection response if the detection confidence is high enough even with partial overlap. In other words, we need a framework that is able to solve for an optimal set of detections by balancing between detection confidences and amounts of overlap.

In this thesis, we propose to use Quadratic Unconstrained Binary Optimization (QUBO) for multi-pedestrian detection. Among a large set of initial possible candidates from a pedestrian detector, we consider selecting the best set of candidates, preferably the optimal set, by optimizing a quadratic objective function containing unary and pairwise spatial terms. This approach reasons directly over the space of overlapping object detections, represented as bounding boxes, by formulating a quadratic objective function that contains both unary scores measuring the quality of an individual detection, and pairwise scores measuring the joint compatibility of pairs of overlapping detections. Loosely speaking, the unary scores reward candidate boxes with high detector confidence, whereas the pairwise scores impose a penalty for excessive amounts of overlap between two boxes. The problem is to find a set of detections that maximizes the quadratic objective function, which is a classic problem of quadratic unconstrained binary optimization (QUBO). Although this is an NP-hard problem, efficient approximate methods are applicable that yield high quality solutions on large problem sizes.

This work shows that quadratic optimization for reasoning about overlapping detections in the case of occluded persons can improve the performance of a pedestrian detection system, especially in scenes containing multiple persons and partial occlusions. The framework is not limited to a specific kind of detector and should be applicable to other kinds of detection as well. It is also flexible enough to be further applied to multi-body-part detection or tracking.

There are various crowd analysis applications where we can apply our frame-
work. For example, we can get an approximate count of the number of people in a frame or over a period of time in a given area. This number and location of people can further be used to analyze the spatial layout of a crowd. Another useful application is flow analysis, where the trajectory of each detected person is tracked over a period of time. The flow of each person and of a whole crowd then can be analyzed for further information about crowd size, motion, and behavior.

1.2 Overview of Optimized Detection Framework

We formulate an optimization-based detection framework that chooses a subset of detections from a large set of possible detection candidates. This subset should be the set that best represents all the pedestrians in an image. The framework solves for a detection solution by balancing between detection confidence scores and amounts of overlap among detection candidates. Specifically, our framework selects a subset of objects from a set of possible detection candidates in order to optimize an objective function that balances unary detection confidence scores against pairwise candidate overlap penalties.

The optimized detection framework consists of two parts; candidate detection and optimization. As shown in Figure 1.5, in the first part, we need to generate a large set of possible detection candidates. Then, we use an optimization method to select an optimal set among those detection candidates that best represent all persons in the image. Later, we consider our optimized detection framework to extend to a data association problem.
1.2.1 Candidate Detection

The main purpose of our detection phase is to generate a set of detection candidates, preferably many of them as shown in Figure 1.5, such that low confidence candidates that might belong to an occluded person are also included. Detectors are not restricted to only sliding-window-based human detectors as long as each detection is associated with a detection score indicating how confident we are that it represents a person. For example, we can also generate detection candidates using a foreground covering method as well.
Our optimized detection framework is flexible enough to be applied to several candidate generation scenarios. We propose three implementations; full body detection, multi-part detection, and multi-frame data association. In full body detection, the detection candidates represent full bodies of a person. For multi-part detection, the detection candidates represent different parts of a body. For example, we can use heads, torsos, and legs as three sets of parts-based detection candidates. The implementation for data association is briefly explained in Section 1.2.3 and in more detail in Chapter 5, but basically detection candidates are generated from two consecutive frames as well as matching pairs of candidates.

1.2.2 Optimization

The goal of the optimization phase is to find a subset of detection candidates that best represent all persons in the image. Each detection candidate is associated with a detection confidence score which will be called a unary confidence score or a unary score. For pairs of detection candidates that overlap with each other, we determine the amount of overlap between them and define pairwise overlap scores or pairwise scores.

The unary scores indicate how likely it is that each candidate is a person so we define these values as (positive) rewards. On the other hand, pairwise scores indicate the amount of overlap between candidates, which could be an overlap due to redundant hypotheses on the same person or an overlap due to an occluded person. We want to allow some overlap (as for the occluded person) but not too much overlap (as an overlap on the same person), so we define these pairwise overlap values as (negative) penalties. The penalties prevent the system to select too many candidates that have a high overlap among them, whereas rewards encourage the
system to select more and higher confidence candidates.

The optimization phase finds a set of candidates that maximize the objective value consisting of rewards and penalties. The framework will attempt to find a balance for occlusion, because allowing too much overlap leads to false positives and allowing too little overlap leads to false negatives or missed detections. This concept leads us to formulate the detection problem as a binary quadratic optimization model.

Quadratic optimization models are used in many applications and various approaches have been proposed to solve these problems. In this work, we evaluate two heuristic approaches and two non-heuristic approaches. The heuristic approaches we use are greedy forward search and Tabu search. For non-heuristic approaches, we use quadratic programming.

Figure 1.6: Concept of framework extension to tracking. (a) Matching candidates (b) Optimized solution with matching shown in the same color.
1.2.3 Extension as a Data Association Framework

A useful application in pedestrian crowd scenes is flow analysis. The spatial displacement of each detected person will be tracked over a period of time to generate a series of flow vectors that traces out a trajectory. The flow of each person or the flow of a whole crowd can be analyzed for further information about motion, interaction, and behavior. We extend our optimized detection to consider detection in pairs of frames so that it will be more applicable in tracking applications and to improve the detection results as well.

We address object tracking as a problem of finding matching objects between pairs of frames. Detected objects in one frame are matched to the detected objects in the next frame. The connected links of those objects between two consecutive frames are called “tracklet”, which is a segment of a complete trajectory. For our framework, we need to find a set of possible tracklets. A set of possible tracklets is not all possible links from n to m objects, where n and m is a number of objects in two consecutive frames, but only connected links for objects within a reasonable distance when we assume that the object cannot move too rapidly. This reduces the number of links generated.

We then extend our previous optimization framework to find a subset of tracklets that best represent detections in two consecutive frames and that yield proper matches between objects in the two frames as for tracking. The formulation of optimized detection in a single frame is thus extended to optimized detection in two consecutive frames, which includes solving for a set of tracklets that represents segments of trajectories of all objects in the two consecutive frames.

The approach is shown in Figure 1.6. Two sets of detection candidates from two consecutive frames are shown on the top left and the graphical representation
is shown on the bottom left. We generate a set of candidate tracklets (shown in pink) to connect detections in the two frames. The optimization phase then finds the best detections in each frame as well as the correspondences (matches) between frames.

1.3 Thesis Outline

This section provides an overview of the structure of the rest of the thesis. Chapter 2 provides a review of the significant literatures in pedestrian detection and relating topics. A background in quadratic unconstrained binary optimization (QUBO) and methods to solve this problem are also discussed in detail.

In Chapter 3, we explain the proposed optimized detection framework for detecting full-body pedestrians. The work presented in that chapter was published in CVPR 2013. “S. Rujikietgumjorn and R. T. Collins. Optimized pedestrian detection for multiple and occluded people. The IEEE Conference on Computer Vision and Pattern Recognition (CVPR), June 2013”. In Chapter 4, we introduce an approach to improve the previous optimized detection framework to better handle occluded pedestrians by incorporating reasoning about occluded and unoccluded body parts in to the model. In Chapter 5, we describe an extension of our optimized detection framework to find tracklets spanning two consecutive frames to extend the detection framework toward data association. The last chapter, Chapter 6, summarizes our contributions of this research work and discusses possible future research directions.
Related Work and Background

2.1 Related Work

In this chapter we review previous work related to pedestrian detection. We also present background material on algorithms commonly used to solve binary optimization problems. Object detection has been widely studied in computer vision, and in recent years, several approaches have been proposed for detecting and tracking pedestrians. In this section, we survey research works for pedestrian detection, with a particular emphasis on crowded scenes with occlusions.

Sliding window based detection, a method that uses a pre-trained classifier to scan the whole image to generate detector responses, is one of the most widely used approach in object detection. Early work on sliding window detectors is presented by Papageorgiou and Poggio [53] where a descriptor is based on nonadaptive Haar wavelets and a support vector machine classifier. It has become a basic approach that has been adapted and extended to several other methods using different features or classifiers.

Large performance gains came with the adoption of gradient-based descriptors.
Dalal and Triggs [8] propose state-of-the-art human detection method using the Histograms of Oriented Gradients (HOG) descriptor. Enzweiler and Gavrilla [15] compared this HOG-based approach with other methods on large datasets including more than 20,000 images and showed that it outperformed the other methods.

Various other types of features have also been used to represent pedestrian appearance templates. Wu and Nevatia [74] use edgelet part features for human detection. The template-based method has also been improved by using more novel features such as shapelet features [62], Granularity-tunable Gradients Partition (GGP) [45], or a combination of informative channels [11]. Yang et al. [79] use a deep neural network method to learn new video-specific features as a boost to detectors performances. Although, additional information such as texture and color can be added to the features, it results in an increasingly high-dimensional feature space. Nevertheless, a dimensionality reduction method called Partial Least Squares (PLS) analysis is adopted by Schwartz et al. [64] to reduce the feature space size while maintaining good performance. Partial Least Squares is a common technique and it dates back to Herman Wold in "Partial least squares" [35].

Since sliding window based methods usually generate multiple overlapped detections on the same person, a widely adopted process is to apply non maximum suppression (NMS) to remove multiple responses from the detectors [8, 23]. Several approaches of non-maximum suppression have been proposed [48]. However, since NMS aims at removing multiple responses at neighboring locations, it often results in missing an occluded person by incorrectly removing the response of that person.

Besides template based approaches, other approaches use low level features such as foreground masks, commonly generated from background subtraction, to hypothesize the locations of objects. Motion detection, using image differencing,
provides an alternative approach to generate foreground pixels [71]. Although foreground masks are widely used in pedestrian detection, are able to identify the area of an object quite well, and can be produced computationally fast, a further segmentation step is required to divide the foreground blobs into each individual person. Several papers in pedestrian detection try to segment and classify the foreground blobs into human body regions using MCMC-based optimization [81], EM formulation [58], or implicit shape models [32]. Segmenting foreground blobs becomes harder when multiple persons are close together such as groups in a crowd. Another drawback of this approach is that it is difficult to differentiate between moving people and other moving objects such as bicycles or cars, since all moving objects are represented as blobs of foreground pixels.

Occlusion is a common occurrence in pedestrian video from most real world scenes. Dollar et al. [12] note the importance of occlusion by analyzing six large well known datasets and show that pedestrian occlusion occurs in more than 70% of the videos, with most of those occlusions occurring in the lower body parts. Thus, the ability to handle occlusion is important, and yet it is difficult to achieve, especially for a dense crowd.

Pepikj et al. [54] focus on modeling spatial occlusion patterns by trying to detect reoccurring arrangements of occluded objects. Their approach was performed on cars, which tend to have a specific pattern of occlusion due to lines of cars parking on the sides of streets. With pedestrians being much smaller in size, the occlusion patterns can be more complicated than that of cars. Other authors try to model the mutual visibility relationship of pedestrians [51]. Ouyang and Wang [50] propose a two-pedestrian detector to detect a pair of pedestrians, where one is being occluded by the other. This approach is limited to two-person occlusion and various models have to be learned for different patterns of occlusion.
Several pedestrian detectors use a full body as a template [8, 15, 64]. However, full body templates are not robust to occlusion, where parts of the body are not seen, and usually result in weak detection responses. Therefore, several approaches use part-based detectors in order to improve detection performance. Felzenszwalb et al. [18] propose a deformable part model with a spatial model allowing small displacements of each part. Because of the relaxation of links between each body part, the deformable layout allows the detector to detect various kinds of poses and is applicable to other categories of objects as well. With flexible layouts, the multi part model can be used to represent various categories of objects such as a person, car, bicycle or chair. For each object categories, different layout of the model can be learned to represent different view or pose [14, 17, 49, 67].

In fact, part-based detectors can be used to handle occlusion more efficiently than full body detectors since they may provide significant cues when partial occlusions occur. For example, when only the lower torso and legs are occluded, the head and upper torso may be clearly detectable. Since most occlusions occur at the lower part of the body, works such as [3, 39, 59] try to handle occlusion by relying only on head detection and ignore other parts of the body. These approaches may work well with a very dense group or crowd, where only each occluded person’s head is clearly visible, but other body parts are also likely to benefit, in general, when partial body information is included. Girshick et al. [25] propose a grammar model of occlusion that can handle scene occlusions (person being occluded by other objects).

Several approaches combine local part-based and global template-based detectors together [33, 41, 66, 75]. Schwartz et al. [63] integrate a face detector with full body detector to better handle occlusion. Their results show that integration has advantages during occlusion over using an individual detector, and therefore,
combining multiple observations can improve detections under occlusions.

In several papers, part detectors are applied to scenes with a small number of people, aiming to detect a single person or multiple people with little or no occlusion [19, 25]. Directly applying part detectors to a group of pedestrians with heavier occlusion does not yield much better results when compared to full body detectors, as the results in [12] show. Their evaluation results show that part-based detections still cannot handle occlusion effectively and that performances of part based detectors degrades as severely as other approaches. A better approach should be used to integrate part-based detectors into a framework that is able to reason about the space of occlusion.

A large literature for body part detectors also lies in the area of pose estimation for gesture and movement analysis [55, 69, 72, 78] which aims at fitting multiple parts to a target person. Part fittings require that each part, such as upper arms, lower arms, upper legs, or lower legs, is accurately placed onto the body to aid analysis of gestures or activities. Since pedestrian detection aims at locating the full person from relative low resolution images and does not require very accurate localization of each part, these methods may not directly be well applicable to a group of occluded persons.

Many pedestrian detection works exist for crowd analysis. Yan et al. [76] propose to globally model the crowd using appearance and spatial interactions by formulating the problem as a maximizing a posteriori (MAP) solution. Rodriguez et al. [59] integrate crowd density estimation to improve detection and tracking in a crowded scene. Rodriguez et al. [60] also proposes a crowd analysis framework for very high density crowd using motion patterns as priors. Since high density crowds usually contains less image resolution compared to pedestrian scenes with low to medium partial occlusion, these frameworks need more information such as
density or motion priors to cope with the lack of appearance information.

Counting number of people in a crowd is another popular task that is difficult when there is a high amount of occlusion. There are several works for crowd counting such as methods based on regression of low-level features [6], clustering trajectories [56], marked point processes [22], or head-shoulder detection with foreground masks [38]. Lempitsky and Zisserman avoid the counting-by-detection method, where count accuracy depends on the performance of detectors, and propose a density-based approach instead [37].

Pedestrian detection systems are often incorporated into a framework for detection-then-tracking [2, 40]. The detection results on each frame are matched to the detection results in the next frame to create trajectories. Brendel et al. [5] formulate the data association problem as a maximum weight independent set problem that solves for tracklets, starting with linked detections from two consecutive frames and using an iterative process to build longer tracks and handle long term occlusions. Several works incorporate context cues like appearance and spatial information as an improvement [10, 65, 77]. These additional information also provides important constraints that can help improve the detection or tracking performance.

Occlusion also occurs commonly in general object tracking. Several works track multiple points of an object such that when a partial occlusion occurs, visible points can still be tracked. Grabner et al. [28] use local image features as supporters to predict the location of an object even when parts of an object are occluded. This multiple tracking points concept can be applied to a person as well. Wu and Nevatia [75] propose to use a human representation as an assembly of four body parts in order to handle tracking pedestrians with both inter-object and scene occlusion. Shu et al. [66] adopt the part-based model for a multi target tracking system to handle partial occlusion in both detection and tracking stages.
Vision tasks that involve balancing tradeoffs, for example, the tradeoff between maximizing coverage while minimizing redundancy, can be rigorously formulated as optimization problems. The object detection framework has been recently formulated as an optimization problem in [9, 20, 59, 61]. Desai et al. [9] introduces a unified model for multi-class object recognition. The overlap between different types of related objects is used as a spatial interaction term in the objective function for object detection. Rodriguez et al. [59] applied a similar framework to crowd scenes, but also integrate crowd density as extra information to their detector. Both of those approaches use a greedy approach to search for an approximate solution. Other methods for solving multi-person detection problems in computer vision include MCMC [22] and Tabu search, which is a meta-heuristic approach that has been used to solve multi-assignment problems in Intelligent Visual Surveillance [13].

Besides detection, several papers in computer vision also employ frameworks involving a combinatorial optimization problem. Trinh et al. [70] propose a hand tracking framework based on binary quadratic programming. The relaxation of the problem is solved then the branch and bound method is used to find a binary solution. The branch and bound technique is widely used to solve optimization problems in various fields [7, 29, 31, 44], however it can be computationally expensive. Amberg and Vetter [1] propose a combinatorial optimization framework for optimal facial landmark detection that selects a globally optimal set of candidates corresponding to shape models using the branch and bound method. Leibe et al. [36] formulate the coupling of detections and trajectory estimations by jointly optimizing them as a quadratic boolean problem.

Current progress in pedestrian detections can be found in [12], where Dollar et al. evaluate 16 state-of-the-art pedestrian detectors on six well known datasets and
shows that even though the results are promising, there is still need for improvement. There are several factors and variations in real world images that can reduce the performance of a detector. Occlusion is one of the research needs suggested by Dollar et al. since their evaluation results show a rapid decrease in performance even with mild occlusion. Surprisingly, for partial occlusion, the performance of the part-based detectors decrease as much as others. Developing a more accurate and more robust pedestrian detector, especially for partially occluded pedestrians, is still an area of great interest.

2.2 Quadratic Unconstrained Binary Optimization (QUBO)

Binary optimization (or 0-1 integer programming) is the problem of finding a binary vector $x = [x_1, x_2, \ldots, x_n]$ that maximizes an objective function $f(x)$. Binary variables can be used to model yes-no decisions, or keep-discard decisions. The objective function is typically represented by a multi-linear quadratic polynomial expression of degree 1 (linear), 2(quadric) or possibly higher order. In this work, we use a quadratic objective function written in the following general form:

$$f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_i x_j + \sum_{i=1}^{n} c_i x_i$$ (2.1)

where each $x_i$ and $x_j \in \{0, 1\}$, each $c_i$ is a unary coefficient, and each $c_{ij}$ is a pairwise coefficient.

The goal is to assign 0,1 values to each decision variable $x_i$ in a way that maximizes the objective function $f(x)$. Noting that for binary variables, $x_i = x_i^2$, we can combine all the coefficients $c_i$ and $c_{ij}$ into a single square matrix $Q$ and
solve the binary integer maximization.

\[ \max_{x \in \mathbb{B}^n} f(x) = \max_{x \in \mathbb{B}^n} x'Qx \]  

(2.2)

The restrictions on the values of the decision variables in an optimization model are called constraints (which are restrictions and/or requirements that specify possible values of decision variables). If the decision variables do not have such constraints, the problem is called an unconstrained optimization problem.

The Quadratic Unconstrained Binary Optimization (QUBO) can be used to formulate different kinds of problems such as maximum vertex packing, minimum covering, or maximum independent weighted sets [34]. Although the QUBO problem is known to be an NP-hard problem [52], efficient approximate methods are available that yield high quality solutions on large problem sizes.

\section*{2.3 Solving QUBO Problems}

Combinatorial optimization problems are, in general, very difficult to solve. These discrete optimization problems do not have optimality conditions that enable us to determine whether the solution found is globally optimal. Solve for a globally optimal solution for a general combinatorial optimization problem by enumerating all possible solutions [57] is not feasible for large problem. As the number of variables increases, the amount of time to solve the problem tends to grow exponentially. Suppose \( x \) is a binary variable and we want to solve for \( x \) that yields the maximum value of a function \( f(x) \), we need to evaluate \( 2^n \) possible solutions. If we are able to evaluate 10 billion solutions per second, it would take about 1 day to solve the problem for \( n = 50 \) and 3.7 years for \( n = 60 \). This is often referred to
as combinatorial explosion.

While the problem takes an enormous amount of computational time to solve to optimality, heuristic approaches are often applied to seek solutions in a feasible computational time. However, these heuristic approaches are not guaranteed to find the globally optimal solution. For example, local search method look for better solutions in a small neighborhood of a current solution. In this regard they are like a blind man trying to find the highest point on an island by keeping exploring a nearby location: once reaching the top of a hill, how can he tell whether this is the highest point on the island?

In our experiments, we examine various approaches for solving the Quadratic Unconstrained Binary Optimization problem. Both heuristic and non-heuristic approaches are explored. For heuristic approaches, we decide to use a greedy method since it seems computationally fast and is widely used in solving optimization problems. Another approach we use is a local search procedure with intelligent decision making strategies called Tabu search. Besides heuristic approaches, quadratic programming are also evaluated. The following section explains each method in more detail.

### 2.3.1 Greedy Forward Search

Greedy algorithms recursively build up a solution by adding one candidate at a time and always greedily chooses the next candidate that yields the most benefit. In this work, we use greedy forward search as in [9, 59] since our problem formulation is comparable.

Greedy Forward Search starts with selecting the candidate that gives the highest cost function. Then, giving the set of previously selected candidates, each
**Algorithm 2.1** Greedy Forward Search

\[ V \leftarrow \text{all candidates}, \ S \leftarrow \emptyset \]
\[ y \leftarrow 0, y' \leftarrow 0 \]

**repeat**

\[ x_k \text{ is a candidate in } V \text{ that yields the highest cost function when the solution is } S \cup \{x_k\} \]
\[ X \leftarrow S \cup \{x_k\} \]
\[ y' \leftarrow f(X) \]

**if** \( y' \geq y \)** then**

\[ S \leftarrow S \cup \{x_k\} \]
\[ V \leftarrow V - \{x_k\} \]
\[ y \leftarrow y' \]

**end if**

**until** \( y' < y \)

remaining candidate is tested one at a time and the one that yields the highest objective function score when added to the current solution set is chosen. The process continues until the next selected candidate reduces the value of the cost function. This simple method has been shown to perform well in practice in many applications. An outline of the greedy forward search method is given in Algorithm 2.1.

In general, greedy algorithms do not guarantee finding an optimal solution. They may be trapped at a local optimum, and may consistently fail in some cases. Nevertheless, they are useful because they are very easy to develop and quick to run, and are often able to give a good approximation to the optimal solution. Greedy algorithms not only have the advantage of a quick solution improvement, but also often terminate with a locally optimal solution after only a small number of iterations. However, this quick termination could also be a drawback when the method cannot improve upon the current solution.
Algorithm 2.2 Tabu Search
\[
x \leftarrow \text{FindInitialSolution()}
\]
\[
\text{TabuList} \leftarrow \emptyset
\]
repeat
\[
\hat{x} \leftarrow \text{BestMove}(N(x)) \quad \triangleright N(x) \text{ is the neighborhood of a solution } x
\]
if \(\hat{x} \notin \text{TabuList}\) then
\[
x \leftarrow \hat{x}
\]
else if \(\hat{x} \in \text{TabuList}\) and Aspiration Criteria met then
\[
x \leftarrow \hat{x}
\]
else
\[
\text{Do not update } x
\]
end if
\[
\text{UpdateTabuList}(x)
\]
until Maximum Iterations Reached

2.3.2 Tabu search

Tabu search is a metaheuristic approach proposed by [26, 30]. It is a common metaheuristic method used for operations research problems [57]. This approach incorporates randomization and uses search memory to help select the next candidate. Using these strategies, Tabu search is often able to avoid being trapped at a local optimal solution. Tabu search has been shown to be an efficient approach for handling combinatorial optimization problems of a large size, and has been successfully used for applications include machine scheduling, graph partitioning, multi-item inventory planning, and call routing in communication [27, 52]. Tabu search is able to find a good solution for several NP classes of problems [73].

An important concept of Tabu search is the use of adaptive memory; starting with an initial solution point, the method sequentially adjusts the value of a small subset of variables in an attempt to find an improved solution in a local neighborhood while maintaining a Tabu list to make sure that previously searched locations are not soon revisited. The Tabu search algorithm is outlined in Algorithm 2.2

Neighborhoods of a solution in discrete spaces are solutions that are slightly
different or very close. For example, neighbors of a binary solution [0 1 1 0 0] can be [0 1 1 0 1] or [1 0 1 0 0]. As another example, consider a traveling salesman problem, when the current selected path is 2 → 5 → 3 → 4 → 1 A neighbor of this solution can be a swap in the path such as 2 → 5 → 1 → 4 → 3.

Two key aspects of Tabu search are intensification and diversification strategies. Intensification encourages Tabu search to search the feasible solution region more thoroughly while diversification allows Tabu search to explore further away in a different region. When the search finds a locally optimal solution, elements are kept in a Tabu list for a period of time according to Tabu tenure in order to prevent revisiting solutions. Elements in the Tabu list are ineligible for consideration, so longer tenure will provide more diversification while shorter tenure will provide more intensification.

### 2.3.3 Quadratic Programming

Techniques for solving continuous optimization problems do not directly help us solve combinatorial optimization problems where variables in the solution must be integer values [57]. Even though discrete problems have fewer number of values than continuous problems, they are harder to solve, which may, at first, seems counterintuitive. However one approximate approach to solve a discrete problems is to solve a continuous relaxation of the problem and then adjust the continuous solution vector to be a binary one.

Given an optimization problem

\[ P = \max \{ f(x) : x \in S \} \]
a relaxation $P_R$ of $P$, given by

$$P_R = \max \{g(x) : x \in S'\}$$

is one where every feasible solution of $P$ is also feasible to $P_R$ so that $S \subseteq S'$ and every feasible solution $x$ in $P$ has a value no worse than $x$ in $P_R$.

Thus, a binary optimization $P$ and its relaxation $P_R$ as a continuous optimization problem are defined as

$$P = \max \{f(x) : x \in \{0, 1\}\}$$
$$P_R = \max \{g(x) : x \in \mathbb{R}\}$$

We can then apply quadratic programming to solve the relaxed problem. For quadratic programming, we use the trust-region method to find the maximization of the continuous version of the original binary problem. The basic idea of the trust region method is to approximate the objective function with a simpler function in a neighborhood called the trust region and to seek a solution to that new trust-region subproblem [47, 57].

Specifically, given a function $f$, a Taylor-series expansion of $f$ around $x_k$ is

$$f(x_k + p) = f_k + g_k^T p + \frac{1}{2} p^T \nabla^2 f(x_k + tp)p$$  \hspace{1cm} (2.3)$$

where $f_k = f(x_k)$ and $g_k = \nabla f(x_k)$, and $t$ is a scalar in the interval $(0,1)$. A quadratic model function $m_k$ is used at each iterate $x_k$. By using an approximation $B_k$ to the Hessian in the second-order term, $m_k$ is defined as follows:

$$m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p$$  \hspace{1cm} (2.4)$$
For each iteration, we solve for a solution of the subproblem

$$
\min_{p \in \mathbb{R}^n} m_k(p) = f_k + g_k^T p + \frac{1}{2} p^T B_k p
$$  \hspace{1cm} (2.5)

s.t. $\|p\| \leq \Delta_k$

where $\Delta_k > 0$ is the trust region radius. For a given step $p_k$, the ratio $\rho_k$, whose numerator is the actual reduction and the denominator is the predicted reduction, is defined as

$$
\rho_k = \frac{f(x_k) - f(x_k + p_k)}{m_k(0) - m_k(p_k)} \hspace{1cm} (2.6)
$$

An outline of the trust region method [57] is given in Algorithm 2.3.

Since the solution is continuous rather than binary valued, one straightforward way to complete the solution is to apply a rounding function to yield a binary value. However, a preferable way to proceed is to use a branch and bound method, which is explained in Section 2.3.4.

### 2.3.4 Branch and Bound

A commonly used method for solving combinatorial and binary optimization problems is the branch and bound algorithm [57], which attempts to eliminate as many of the feasible solutions as it can without actually evaluating them. Branch and bound algorithms divide the feasible region into smaller subregions, and then examine each subregion for the best integer solution. A continuous relaxation of the integer problem is often used to determine bounds on the integer solution to decide which subregion should be further examined. This approach allows solving large problems with less work.

Let us define some definitions for branch and bound here:
Algorithm 2.3 Trust Region Method

Given $\Delta > 0$, $\Delta_0 \in (0, \Delta)$, and $\eta \in [0, \frac{1}{4})$:

for k=0, 1, 2, ... do
  
  Find $p_k$ by approximately solving Eq.2.5
  
  Find $\rho_k$ from Eq.2.6
  
  if $\rho_k < \frac{1}{4}$ then
    
    $\Delta_{k+1} = \frac{1}{3}\Delta_k$
  
  else
    
    if $\rho_k > \frac{3}{4}$ and $\|p_k\| = \Delta_k$ then
      
      $\Delta_{k+1} = \min(2\Delta_k, \Delta)$
    
    else
      
      $\Delta_{k+1} = \Delta_k$
    
  end if
  
  end if
  
  if $\rho_k > \eta$ then
    
    $x_{k+1} = x_k + p_k$
  
  else
    
    $x_{k+1} = x_k$
  
  end if
  
end for

Branching Dividing the current subregion into smaller ones.

Branching Variable A variable used for decomposing the region.

Fathom a Node A node is fathomed when we do not branch from that subproblem either because its bound is worse than the current best-known solution or because of infeasibility. Since a fathomed node is not expanded further, any subregions of that node do not have to be considered, leading to a computational saving.

Active Nodes A node that has not been branched or fathomed.

Incumbent Solution A current best integer solution.

A subproblem will not be branched when it is fathomed or it yields an integer solution. When all other nodes have been fathomed and there are no active nodes, the current incumbent solution is the optimal solution to the optimization problem.

A general branch and bound algorithm [57] for an integer problem is shown in
Algorithm 2.4.

Algorithm 2.4 Branch and Bound

Step 0: Find an Initial Solution
Solve the relaxation problem for an initial solution. If all variables $x_i$ are all integers, the current solution is optimal and we are done. Otherwise, go to Step 1.

Step 1: Select a Branching Variable
Choose one variable, $x_k$, that is not an integer, $v_k \notin \mathbb{Z}$, to be the branching variable.

Step 2: Formulate Subproblems
Create two subproblems where one subproblem has constraint $x_k \leq \lfloor v_k \rfloor$, while the other has constraint $x_k \geq \lceil v_k \rceil$. Then, solve each of these subproblems.

Step 3: Test for Integer Solution
Check both solutions from subproblems in Step 2 to see whether the corresponding solution is an integer. If not, go to Step 4. If yes, a feasible solution is found. If its objective function value is better than our current best solution, store it as our incumbent solution. Then, for every active node, compare its relaxed bound to the new incumbent solution. If the relaxed bound is not better, fathom the node.

Step 4: Check Bound Against the Incumbent Solution
If there is no incumbent solution, go to Step 6. Otherwise, compare the relaxed bounds from each subproblem in Step 2 with the current incumbent solution value. If the relaxed bound is not better, fathom the node.

Step 5: Test for Infeasibility
If the subproblems in Step 2 is not feasible, fathom the node.

Step 6: Select a Subproblem
If there are still active nodes, select one of them to further evaluate and go to Step 1. If there are no more nodes to evaluate, we are done and our current best solution is the optimal solution.

Using Algorithm 2.4 with binary optimization problems, the Branch and Bound method is greatly simplified because branching variables are binary, so subregions formed by branching on $x_i = 0$ and $x_i = 1$. An example shown in Figure 2.1 illustrates a problem of finding binary values for $x_1, x_2, x_3$, and $x_4$ that yields the highest cost function.

1. Define subproblem 1 as an initial solution $x = [1, 1, 0.5, 0.5]$ with cost function
$z$ equals 44.

2. Choose $x_3$ as a branching variable. Arbitrarily choose subproblem 3 to solve. A solution is [1,0,7,1,0] with $z$ equals 43.7.

3. Choose $x_2$ as a branching variable. Solve subproblem 4. A solution is [1,0,1,1] with $z$ equals 36. Since the solution is binary, subproblem 4 yields an incumbent solution with current lower bound equals 36.

4. Solve subproblem 5. A solution is [0.6,1,1,0] with $z$ equals 43.6.

5. Choose $x_1$ as a branching variable. Subproblem 6 yields an incumbent solu-

Figure 2.1: An example illustrates steps for a branch and bound to further divide a subproblem or fathom a node, depicted as a red cross.
tion \([0,1,1,1]\) with \(z\) equals 42. Thus, fathom subproblem 4 and update the lower bound to 42.

6. Subproblem 7 yields a binary solution, but \(z\) is less than current bound so it is fathomed.

7. Go back to solve for the remaining node which is the subproblem 2. Then choose \(x_4\) as a branching variable.

8. Subproblem 8 yields a binary solution, but \(z\) is less than current bound so it is fathomed.

9. Solve subproblem 9 then choose \(x_2\) as a branching variable.

10. Fathom both subproblem 10 and 11 because both \(z\) values are less than the current bound.

11. The incumbent solution from subproblem 6 is the optimal solution.
Chapter 3

Optimized Pedestrian Detection for Multiple and Occluded People

3.1 Introduction

In this chapter we introduce our quadratic binary optimization framework for optimization-based pedestrian detection and develop the framework for finding full-body pedestrian hypotheses. Later chapters adopt this approach to multi-body-part hypotheses, and two frames detect-and-track hypotheses.

We propose an approach that reasons directly over the space of overlapping object detections, represented as bounding boxes, by formulating a quadratic objective function that contains both unary scores measuring quality of an individual detection, and pairwise scores measuring the joint compatibility of pairs of overlapping detections. Loosely speaking, the unary scores reward candidate boxes with high detector confidence, whereas the pairwise scores impose a penalty for excessive amounts of overlap between two boxes. The problem is to find a set of detections that maximizes the quadratic objective function, which is a classic
Figure 3.1: An example of optimized detection. A dense set of the candidate detections is generated in (a) and (c), then a binary quadratic optimization procedure is applied to choose the best set of detections that maximizes the tradeoff between a detection confidence score and an overlap penalty. (a) Candidate detections from the shape covering method. (c) Candidate detections from the sliding window human detector. (b)(d) The final set of optimized detections are shown in yellow, and false negative (missed detections) are shown in red.

Our approach is a framework that selects a set of objects from a discrete set of possible candidate detections to optimize an objective function that balances unary confidence scores against pairwise overlap penalties. An example of our framework is shown in Figure 3.1. First, a dense set of possible candidates (Figure 3.1(a)
and 3.1(c)) is generated by two instantiations of our approach; one hypothesizes candidates based on a shape covering over a binary foreground mask, whereas the second uses a standard appearance-based pedestrian detector without non-maximum suppression. The goal is to determine which subset of these multiple overlapping detections best represents the number, location, and shape of people in the image. The object configuration that maximizes our quadratic objective function is found and shown in Figure 3.1(b)(d). The aim of this paper is to show that the quadratic optimization for reasoning about overlapping detections can improve the performance of a pedestrian detection system, especially when there are multiple, overlapping persons.

### 3.2 Optimized Detection Framework

We proposed a framework that uses Quadratic Unconstrained Binary Optimization (QUBO) for pedestrian detection in crowded scenes. Our framework consists of two parts. The first part is to generate a dense set of detections that contains many possible detection candidates. Each candidate is a bounding box or ellipse shape that may or may not outline a person in the image. Two different approaches for generating candidates are used: foreground shape covering and detector confidence map filtering, which are explained in Section 3.2.1. The second part is a binary quadratic optimization procedure that searches for an assignment of 0’s and 1’s to candidate ellipses that yields a high, ideally the maximum, objective function value.

Figure 3.2 presents a “big picture” overview of how our approach would be incorporated into a typical pedestrian detection pipeline. First, an existing pedestrian detector is applied to produce either a detection confidence score map, or if
that is not available as an output, a set of unfiltered bounding boxes with associated confidence scores, produced with a low confidence threshold. Our method then samples a large but finite set of plausible candidates, and associates a unary confidence score with each one. A pairwise score is also computed for each pair of overlapping candidates, to specify the penalty that would be incurred if both candidates are kept in the final solution. The (positive) unary and (negative) pairwise scores form a quadratic objective function, and a binary quadratic optimization procedure is performed to search for an optimal configuration of detections to keep, as determined by the objective function value.

### 3.2.1 Generating Dense Detection Candidates for Pedestrian Detection

The proposed framework can be incorporated into a typical pedestrian detector that generates many possible detection candidates each with a detection confidence score. We apply our framework to two different detection approaches to demonstrate the ability to work with different types of detectors. The first approach is a shape covering method that uses foreground mask data. The second approach uses an off-the-shelf appearance-based pedestrian detector that is based on a histogram of oriented gradient (HOG) feature representation.

**Detection Candidates by Shape Covering**

Previous works have considered the problem of detecting people as a “shape covering” of foreground mask data [22, 80]. That is, given a foreground mask computed by either background subtraction or motion analysis, a solution is sought as to the number, location, size and possibly articulation of a set of shapes to cover as many
foreground pixels as possible, while leaving as many background pixels as possible uncovered. To avoid an unnecessary proliferation of overlapping shapes, these methods augment the covering quality term of the objective function with either prior terms on the number of objects present, or with data terms that penalize an excessive object overlap. The previous works [22, 80] used an expensive Markov Chain Monte Carlo stochastic search procedure to find a good shape covering. In this work, we address a similar shape covering problem using QUBO.
To use this shape covering approach in practice, we first generate a lookup table relating location (x,y) in the image to an expected height and width of a pedestrian centered at that location. In our experiments we have created this lookup data from the ground truth camera calibration information; however, in Section 3.2.1 we show how this prior size information can be learned automatically from training data. Given an automatically computed foreground mask, a candidate set of elliptical shapes is generated by methodically sampling midpoint locations every 10 pixels in x and y, looking up the expected width and height at each location, and computing the unary score $c_i$ for each ellipse $x_i$. Each ellipse $x_i$ is assigned a unary confidence score $c_i(x_i)$ computed as

$$c_i(x_i) = \text{on}(x_i) - \alpha \text{off}(x_i)$$ (3.1)

where $\alpha = 0.5$ in all of our experiments and candidates with $c_i \leq 0$ are discarded. Function $\text{on}(x_i)$ returns the number of pixels that are “on” within the ellipse, and $\text{off}(x_i)$ returns the number of “off” pixels. Figure 3.3 illustrates the process of computing a unary score by counting the numbers of on and off pixels.

Figure 3.3: An example of computing unary score from foreground pixels using an elliptical shape mask. The score for this example is 2307.
in an elliptical shape mask. In this example, there are 7063 on pixels and 9512 off pixels. So, using Eq 3.1, the unary score is 2307. Because our objective function will be seeking to maximize sums of unary scores, candidates that have negative unary scores can be immediately discarded. An example set of elliptical shape candidates generated from a foreground mask image is shown in Figure 3.4.

**Detection Candidates by Bounding Box Filtering**

Many object detection approaches employ a sliding window based detector to generate a confidence score map, and then generate a set of final detections through a process of non maximum suppression. For our approach, we modify an existing Histogram of Oriented Gradients (HOG) based pedestrian detector [8], available in OpenCV, and apply it at multiple scales without non maximum suppression to generate a multi-scale detection confidence map as shown in Figure 3.5. A set of initial candidates is then randomly sampled from each detection scale, with a likelihood of sampling being proportional to the detector confidence map values at

![Figure 3.4: A set of elliptical shape candidates in (b) is generated from a foreground image in (a) by the shape covering approach.](image)
each scale level. Figure 3.6(b) shows candidates sampled from one scale level using the confidence score map in (a).

**Learning a Prior on Bounding Box Size**

The camera viewpoint obviously affects the range of scales that can be observed. For example, images taken by a camera near eye level will have a large allowable range of scales, whereas an elevated camera farther away may not see much difference in people size across the image. When camera calibration information is available, such effects can be computed apriori; however, calibration information is often not available. Previous works have used pedestrian height distributions to

![Multi-scale confidence score map](image)

Figure 3.5: A multi-scale confidence score map at four different scales.
Figure 3.6: An example set of candidates generated by detector confidence map sampling. (a) Confidence score map from a single scale level of the multi-scale detector confidence maps. (b) Candidate samples generated from it by random sampling.

compute camera calibration [43, 59].

In this work, we have explored learning size information from dense detection candidates in the form of a lookup table on expected bounding box size versus location in the image. Sliding window detectors that do not have access to such information are prone to a greater number of false positives due to the detections that are either too large or too small. At the very least, having access to minimum and maximum scales at which to expect detections is helpful, yet even this small amount of information can be scene specific and thus troublesome to set by hand.

We employ a simple online learning approach that learns a regression function on size from a set of high confidence detected ellipses containing height information at multiple image locations. Figure 3.7 shows a plot of y coordinate location in an image (depth in the scene) versus height of detected ellipses. When there are multiple heights of people observed over many different image locations, we can compute an approximate height model from this data. The light blue line is a
quadratic regression function learned from one frame only, while the dark blue line is the height approximation found using data from several images taken from a stationary camera view. These approximations can be used to identify outlier detections, represented by green dots, that do not have an appropriate size. Rather than apply a hard threshold to filter out these improperly sized detection, we reduce the unary confidence score in our approach according to the dissimilarity between a candidate bounding box scale and the expected detection height at that location. The detection confidence score will be greatly reduced when the detected ellipse is much larger or much smaller than the learned size estimate. This simple procedure penalizes detections with improper scales, reducing the chances that they will be kept in the solution vector returned by QUBO.

### 3.2.2 Formulation as a QUBO Problem

Binary optimization is the problem of finding a binary vector \( x = [x_1, x_2, ..., x_n] \) that maximizes a quadratic objective function \( f(x) \) as explained in Section 2.2. In this work, we use a quadratic objective function written in the following general form:

\[
    f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_i x_j + \sum_{i=1}^{n} c_i x_i
\]  

(3.2)

where each \( x_i \) and \( x_j \in \{0, 1\} \), each \( c_i \) is a unary coefficient, and each \( c_{ij} \) is a \( O(n^2) \) pairwise coefficient.

The goal is to assign 0,1 values to each decision variable \( x_i \) in a way that maximizes the objective function \( f(x) \). Noting that for binary variables, \( x_i = x_i^2 \), we can combine all the coefficients \( c_i \) and \( c_{i,j} \) into a single square matrix \( Q \) and
Figure 3.7: Expected human sizes learned as a regression function between y coordinate location in the image and expected height of a detected pedestrian at that location.

solve the binary integer maximization.

$$\max_{x \in \mathbb{B}^n} f(x) = \max_{x \in \mathbb{B}^n} x'Qx$$  \hspace{1cm} (3.3)

Objective Function

The quadratic objective function in Eq. 3.2 is computed by combining unary and pairwise terms. Each unary score $c_i$ is a measure of confidence that ellipse $x_i$ represents a person. On the other hand, the pairwise scores $c_{i,j}$ penalize excessive overlap of pairs of ellipses. All pairwise terms are negative and set to the overlap
ratio, computed as the intersection area (in pixels) of the two overlapping ellipses, divided by the area (in pixels) of the smaller ellipse. If there is only a small overlap penalty, it might be reasonable to keep both candidates. On the other hand, if there is a large overlap, a higher penalty is applied, and it is probably better to remove one ellipse.

Figure 3.8 illustrates an example of cost function computation. In this example, the three ellipses from left to right have unary values 3425, 4412 and 3658 computed from Eq. 3.1. For each pair of distinct overlapping ellipses $x_i$ and $x_j$, we assign an overlap penalty $c_{i,j}(x_i, x_j)$ as -4594 for ellipse $x_1$ and $x_2$, -1998 for ellipse $x_1$ and $x_3$, and -3432 for ellipse $x_2$ and $x_3$. From Eq. 3.3, we want to find a maximal value of $x'Qx$ with the constraint that $x = [x_1, x_2, x_3]$ can take only binary values. We find that the optimal solution $[1, 0, 1]$ specifies that ellipse $x_1$ and $x_3$ should be kept, while ellipse $x_2$ should be discarded. Note that if we applied a traditional, greedy non maximum suppression approach where the ellipse with highest confidence is chosen first while suppressing overlapping ellipses of lesser score, we would have chosen to keep only the middle ellipse $x_2$, while suppressing the other two.

To allow for adjusting the influence of unary versus pairwise scores, the matrix $Q$ in the objective function in Eq. 3.3 can then be formed as

$$Q = w_1 U - w_2 P$$

(3.4)

where $U$ is a diagonal unary score matrix, $P$ is the pairwise score matrix (overlap ratios) and $w_1, w_2$ are relative weights where $w_1 + w_2 = 1$, determined as described in the following section. Both $U$ and $P$ are normalized to be values between 0 and 1.

Furthermore, we can extend Eq. 3.2 by adding a second unary term that rep-
Figure 3.8: Using quadratic binary optimization to find the best shape covering of a foreground mask.

represents the score from a second detector or other additional information. For example, in our experiments we have explored combining unary scores computed from foreground shapes with unary scores computed from a detector confidence map. In this case, the matrix $Q$ is formed as

$$ Q = w_1 U_1 - w_2 P + w_3 U_2 $$

(3.5)

with $w_1 + w_2 + w_3 = 1$.

**Weight Parameter Estimation**

Although both unary and pairwise score matrices contain normalized values between 0 and 1, they represent very different types of information, one being an appearance-based detection confidence and the other being an area ratio. Furthermore, the amount of “acceptable” bounding box overlap for a given situation may depend on the expected density of people in the scene as well as on the camera
viewpoint. For this reason, it is better to weight the relative contributions of the unary and pairwise terms with weighting parameters learned from representative training data. In this section we describe how to learn the relative weights $w_1$ and $w_2$ in Equation 3.4 from training data, constrained so that $w_1 + w_2 = 1$.

The pattern search method is a type of blind search that requires no knowledge of the gradient of a function or even that the function be differentiable [46]. Since it does not need a gradient to determine the direction of search, it can be applied to a nonlinear or discontinuous problem. Pattern search is performed by searching a set of points defined ever a regular pattern or mesh where these points are used to evaluate the function. The pattern or mesh changes its size over iterations according to the search result. If the search returns a point that improves the cost function, the algorithm uses that point as a next iteration point and then reduces the pattern size and defines a new set of points for the pattern or mesh. If no point improves the cost function, the pattern expands to include more points. The search will stop once the pattern reaches a minimum size. We use Pattern Search to find values for $w_1$ and $w_2$ that maximize our objective function. In this case, it is a one dimensional search for $w_1$ since $w_2 = 1 - w_1$. For hybrid approach that combine detector score and number of foreground pixel from shape covering, another unary score is added and the constraint becomes $w_1 + w_2 + w_3 = 1$.

**Solving QUBO**

In this work, we evaluate different methods for solving the QUBO problem: a greedy forward search approach; a metaheuristic method called Tabu search; quadratic programming applied to a relaxed continuous version of the problem. Each algorithm was explained in detail in Chapter 2.3.
3.3 Experimental Results

In this section we evaluate our proposed quadratic binary optimization framework to detect overlapping pedestrians using three approaches for generating candidates and objective function scores: a shape covering of foreground pixels, a method for filtering sampled overlapping detections from a multi-scale sliding window detector, and a hybrid approach of adding foreground covering as a second unary term. We also evaluate three methods for solving the constructed QUBO optimization problems: Tabu Search, greedy algorithm, and quadratic programming.

3.3.1 Evaluation Datasets

We perform quantitative evaluation of our approach using two pedestrian datasets with different viewpoints and scene types. The two datasets are described below and an example image from each dataset is shown in Figure 3.9. To determine whether a detection is a true positive or false positive, we adopt the PASCAL measure for evaluation, which specifies that an area of overlap between a ground truth bounding box and a detected bounding box must exceed 50% and each box can be matched at most once [16]. The result from our framework will be compared against two publically available baseline detectors: the HOG-based detector in OpenCV and the PLS detector[8, 64].

Dataset A

Dataset A is the Terrace sequence from CVLab, Ecole Polytechnique Federale de Lausanne [21]. This dataset is chosen because it has multiple people and the low camera elevation causes a large amount of occlusion, making it a challenging sequence for pedestrian detection. In the current experiment we used 20 frames
as training data and 181 frames as testing data. More sequences are also available from this dataset.

**Dataset B**

Dataset B is a S1.L1 sequence from PETS 2009 which is a popular dataset for pedestrian detection. The camera in this PETS 2009 sequence has a high viewpoint resulting in a small size and similar height of people across the image. This dataset contains a large group of people walking on a street.

### 3.3.2 Results

Figure 3.10 shows a result from each step of our proposed optimized detection approach using two different methods for generating candidate detections. The top row uses shape covering to produce a dense set of candidate detections. The bottom row uses candidates sampled from confidence maps produced by a sliding window human detector. We perform a binary quadratic optimization procedure to choose the best set of candidates that maximize the tradeoff between unary confidence score and pairwise overlap penalty. The final detection result computed

![Figure 3.9: Example images from three evaluation datasets. (Left) dataset A: CVLab’s Terrace sequence dataset; (right) dataset B: PETS 2009 dataset.](image)
by quadratic programming is displayed in Figure 3.10(d) and (h). Yellow detections represent true positives, while red detections indicate a missed detection.

Figure 3.11(a) shows a ROC comparison between Tabu Search, greedy algorithm, and quadratic programming results starting with candidates generated from shape covering. Both greedy algorithm and quadratic programming perform reasonably well. They have lower miss rate and lower false positive rate than Tabu search. The overall accuracy of the greedy method is 0.6849 which is slightly lower than quadratic programming, which has an overall accuracy of 0.6925.

Based on this preliminary test, we decided to use quadratic programming in further experiments. The ROC curves of Figure 3.12 and Figure 3.13 compare results from our three approaches for computing candidates and scores against OpenCV’s

![Figure 3.10: A result from each step of our proposed approach using two different detection approaches for generating candidates. The top row uses shape covering to produce a dense set of candidate detections. The bottom row uses a sliding window human detector. (a) Foreground image. (e) Confidence map from a human detector. (b)(f) Candidate detections exceeding a confidence threshold. (c)(d)(g)(h) Final set of optimized detections shown in yellow ellipses. False negatives are shown in red.](image)
HOG-based human detector and the PLS detector of [64]. The OpenCV’s default HOG-based human detector implementation applies mean shift mode detection as a non maximum suppression method [8]. Recall that we also modified OpenCV’s default HOG-based human detector by not applying non maximum suppression and using it to generate a multi-scale detection confidence map for use in generating candidates by confidence-weighted random sampling. Therefore, this test offers a direct head-to-head comparison of our candidate pruning approach vs non maximum suppression for pedestrian detection based on HOG features.

Figure 3.12(a) shows a ROC comparison among 5 methods using dataset A and Figure 3.12(b) shows a precision recall curve. Our approach 2 and 3 perform better than the rest. Our approach 1 using shape covering performs as well as the HOG-based detector and PLS detector. In fact, this simple approach based on finding

![ROC Curve](image)

Figure 3.11: ROC curve comparison between Tabu search, Greedy, and Quadratic programming using a sliding window human detector to generate initial candidates.
Figure 3.12: Evaluation result of 5 methods (our 3 approaches, OpenCV’s HOG-based detector and PLS detector) on dataset A.
Figure 3.13: Evaluation result of 5 methods (our 3 approaches, OpenCV’s HOG-based detector and PLS detector) on dataset B.
shape coverings of foreground masks works surprisingly well given the simplicity of the approach as compared to the sophisticated appearance-based detectors it is being compared against. Comparing approach 1 and 2, using confidence score from a pedestrian detector yields better results because the unary term based on detection confidence is a better representative score for indicating the presence of a person. This indicates that although our framework is flexible enough to be applied to various detectors or representations, the quality of the unary term score does affect the results. Performance improvement in approach 3 show that we can further improve the results by adding foreground covering as a second unary term. As can be seen, this hybrid approach produces the best end results. Additional information that is likely to further improve the results would be to add crowd density information as in [59].

<table>
<thead>
<tr>
<th>Method</th>
<th>TP</th>
<th>FN</th>
<th>FP</th>
<th>Accuracy</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape covering</td>
<td>515</td>
<td>185</td>
<td>106</td>
<td>0.6390</td>
<td>0.7357</td>
</tr>
<tr>
<td>confidence score</td>
<td>548</td>
<td>152</td>
<td>29</td>
<td>0.7517</td>
<td>0.7829</td>
</tr>
<tr>
<td>confidence score+FG</td>
<td>558</td>
<td>142</td>
<td>21</td>
<td>0.7739</td>
<td>0.7971</td>
</tr>
<tr>
<td>HOG based detector</td>
<td>432</td>
<td>268</td>
<td>14</td>
<td>0.6050</td>
<td>0.6171</td>
</tr>
<tr>
<td>PLS detector</td>
<td>458</td>
<td>242</td>
<td>215</td>
<td>0.5005</td>
<td>0.6543</td>
</tr>
</tbody>
</table>

Table 3.1: Quantitative comparison of the 5 methods on dataset A.

<table>
<thead>
<tr>
<th>Method</th>
<th>TP</th>
<th>FN</th>
<th>FP</th>
<th>Accuracy</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>shape covering</td>
<td>3142</td>
<td>1240</td>
<td>92</td>
<td>0.7023</td>
<td>0.9716</td>
</tr>
<tr>
<td>confidence score</td>
<td>2183</td>
<td>2199</td>
<td>399</td>
<td>0.4566</td>
<td>0.8455</td>
</tr>
<tr>
<td>confidence score+FG</td>
<td>3142</td>
<td>1240</td>
<td>94</td>
<td>0.7020</td>
<td>0.9710</td>
</tr>
<tr>
<td>HOG based detector</td>
<td>1580</td>
<td>2802</td>
<td>106</td>
<td>0.3520</td>
<td>0.9371</td>
</tr>
<tr>
<td>PLS detector</td>
<td>2017</td>
<td>2365</td>
<td>193</td>
<td>0.4409</td>
<td>0.9127</td>
</tr>
</tbody>
</table>

Table 3.2: Quantitative comparison of the 5 methods on dataset B.
Table 3.1 displays quantitative results from the five methods that were evaluated. Our hybrid approach 3 has the highest true positive and accuracy scores. The accuracy, precision, and recall are defined as

\[
\text{accuracy} = \frac{tp + tn}{tp + tn + fp + fn}
\]

\[
\text{precision} = \frac{tp}{tp + fp}
\]

\[
\text{recall} = \frac{tp}{tp + fn}
\]

where \( tp = \) true positive, \( fp = \) false positive, \( tn = \) true negative, and \( fn = \) false negative. Approach 2, 3 and HOG have fewer false negatives than others indicating they have fewer missed detections. Approach 2, 3 also have fewer false positives compared to others. The evaluation result from dataset B is shown in Figure 3.13 and Table 3.2. Our approaches 1 and 3 have more correct detections than the others, and their accuracy and precision are also higher. Figure 3.14 and Figure 3.15 shows several illustrative results of our 3 proposed approaches (shape covering, confidence score, a hybrid of confidence score and covering), OpenCV’s HOG-based detector and PLS detector on both dataset A and B.

### 3.4 Conclusion

We have proposed a framework for improving pedestrian detection performance in cases where there are multiple, overlapping objects. A quadratic unconstrained binary optimization (QUBO) framework is adopted, where a quadratic objective function is formed from unary confidence scores and pairwise overlap penalties. The unary term is not limited to a specific type of detector and can be applied to various types of detector confidence representations, with some adjustment. Solving for the binary solution vector that maximizes our quadratic objective function automatically balances the tradeoff between encouraging multiple, high-quality de-
Figure 3.14: Comparison result of our approach with other methods on dataset A. Yellow means correct detection, red means missed detection, and blue means false positive. (a) Our approach 1: shape covering. (b) Our approach 2: confidence score. (c) Our approach 3: a hybrid combining confidence score and foreground information. (d) OpenCV’s HOG-based detector. (e) PLS detector [64].
Figure 3.15: Comparison result of our approach with other methods on dataset B. Only the correct detections (yellow ellipses) are shown here. (a) Our approach 1: shape covering. (b) Our approach 2: confidence score. (c) Our approach 3: a hybrid combining confidence score and foreground information. (d) OpenCV’s HOG-based detector. (e) PLS detector [64].
tions, while discouraging excessive amounts of overlap. Since the QUBO problem is NP-hard, finding exact solutions for the large-scale problems we generate here is not possible, however, we evaluate three approximate solution methods: a
heuristic Tabu search, greedy forward search algorithm, and trust-region quadratic programming, and found that they all perform well in the experiments, although the last two methods perform the best.

Our qualitative results show that the use of binary quadratic optimization to explicitly reason about detection bounding box confidences and overlaps yields a performance improvement over existing methods in terms of lower miss rates and lower number of false positives. Furthermore, the proposed method can be used to improve the performance of any existing sliding window detector that produces either a detection confidence map or a set of unfiltered bounding boxes with associated confidence scores. We have also demonstrated that the proposed method performs well with a simple detector based on shape covering of a foreground mask. This covering approach can be generalized to use more realistic pedestrian shapes than ellipses, such as those used in [22, 80].
Chapter 4

Improving Occluded Pedestrian Detection by Modeling Unoccluded Body Parts

4.1 Introduction

Occlusion is a common occurrence in pedestrian videos as shown in [12] who analyzed six well-known pedestrian datasets. Among those videos, 53% of the pedestrians are occluded in some frames while 19% of the pedestrians are occluded in all frames. The analysis also shows that several current state-of-the-art detectors experience a reduction in performance when occlusion occurs. Several detectors are able to detect unoccluded persons well, but often miss partially occluded persons. The ability to handle occlusion is important and yet difficult to achieve, especially for a dense crowd. Occlusion makes pedestrian detection difficult because body parts that are not seen can cause weak or missing detections for a detector trained on the full body. Conventional pedestrian detection approaches run a full-body
Figure 4.1: Example cases where a person is occluded but some body parts are clearly visible. (a) visible head, torso, and leg (b) visible head and torso (c) only head is visible.

detector on sliding windows throughout the image, followed by non maximum suppression to remove multiple responses of the detector on the same individual at slightly shifted spatial locations and neighboring scales. However, non maximum suppression can remove a correct cluster of responses when multiple people are close together or partially overlapped, causing a missed detection. Nevertheless, even when a full body hypothesis for an occluded person has a low detection confidence score, the still-visible parts such as head or torso may continue to have high part-based detection confidence scores. Figure 4.1 shows examples where a person in the background is occluded but some of their body parts still are clearly visible. Detecting those visible parts should provide a stronger response for that occluded individual compared to a full body detector, and therefore can improve the detector performance in a crowded scene. Since we aim to improve the performance of a pedestrian detector by improving its ability to detect occluded persons, we propose an approach that uses a multiple-body-parts representation called “body plan” to be able to reason about occlusion. Because a weak detection response
from a full body detector is likely to be suppressed, the body plan representation breaks the body into multiple parts, some of which may or may not be visible in the image. Reasoning about occlusion and overlap then helps to decide how likely a partial body plan is. The detection framework should allow a body plan with a missing part if the detection confidence of the visible parts is high enough and if the missing part can be explained by occlusion with another persons body plan. Basically, the new framework optimizes the tradeoff between part detection confidences, the occlusion of missing parts, and the amounts of overlap between neighboring body plan.

**Contributions in This Chapter**

- We propose a detection framework that reasons over the space of overlap (between detected parts) and occlusion (between missing parts and detected parts) in order to improve the detection of occluded persons.

- We present a search based approach for learning weight parameters. Given a set of parameters, the approach searches for a set of values of those parameters that maximizes the detection accuracy over a set of training images.

**4.2 Related Work**

In recent years, several approaches have been proposed for detecting pedestrians, and significant progress has been demonstrated. Different methods have been proposed to improve the detection of occluded people. [63] integrates a face detector with a full body detector to better handle occlusion. Besides using a full body detector, several approaches use individual body part detectors in order to improve the detection performance [19][69][14][42]. Deformable part based models can also
provide a framework for handling occlusion effectively [25]. Some approaches focus on modeling occlusion patterns [54] while others design a multi-pedestrian detector to handle occlusion [50]. Vision tasks that involve balancing tradeoffs, for example the tradeoff between maximizing coverage while minimizing redundancy, can be rigorously formulated as optimization problems. The object detection framework has been recently formulated as an optimization problem in [20][9][59][61]. The authors of [9] introduce a unified model for multi-class object recognition. The overlap between different types of related objects is used as a spatial interaction term in the objective function of an object detection. [59] applied a similar framework to crowd scenes but also integrated a density of the crowd as an extra information to their detector. Both of those approaches use a greedy approach to search for an approximate solution.

4.3 Detection Candidates

Detection candidates are possible detection solutions for a set of pedestrians in an image. In this framework, detection candidates are body plans. A “body plan” is a combination of multiple body components which may consist of only one or more body parts. The examples in Figure 4.1 illustrate a representation for a three-part body plan consisting of (a) head, torso, and leg, (b) head and torso, (c) head only. A body plan with all body parts is a full body candidate whereas a body plan with a subset of body parts is a weak candidate. These multi-part body plans are used as representations to better represent an occluded person. We use a body-part detector, instead of using a full body detector, in order to increase the detection confidence of the visible parts. Unlike the detector of a visible part, a full body pedestrian detector usually produces a low confidence detection for a
person who is occluded or when some parts of the body are not easy to detect, such as a person wearing clothes having similar color to the background. Section 4.3.1 describes how body part candidates are generated and Section 4.3.2 explains how to combine these body parts to generate body plans.

### 4.3.1 Generating Multiple Body Parts

One state-of-the-art part detector is the Deformable Part Model [19] where an object is modeled as a collection of parts arranged in a deformable configuration. An example of the person model from the Deformable Part Model is shown in Figure 4.2. This model consists of seven body parts. The left image shows responses from part filters. The middle image shows a spatial model for the part location. The right image shows filters specifying weights for histogram of oriented gradients features. We use the Deformable Part Model to generate detections for multiple

![Figure 4.2: An example of the person model. (left) Several higher resolution part filters. (middle) Spatial model for the location of each part. (right) The filters specify weights for histogram of oriented gradients features](image)

body parts. To reduce the complexity, we use a three-component body part model, composed of head, torso, and legs. We group those seven parts into the three body components as shown in Figure 4.3. The head part filter corresponds directly to a head body part in the model. The four middle part filters are grouped into a torso, shown as a green box, and the bottom three filters are grouped into a pair of legs, shown as a blue box. The confidence score of the torso is an average detection score of the four part filters. Similarly, the confidence score of the leg is an average detection score of the three part filters. Other detectors like Harr Cascade detector can also be used to generate multi-part detections. These detectors need to be trained for specific body parts such as head, upper body, and legs. Any type of sliding window based detector could be used as long as the IT can provide a large number of detections, such that a body part of an occluded person is also included as a candidate. Figure 4.4 illustrates an example of a dense set of detection candidates for head, torso, and legs generated from our modified Deformable Part Model.
4.3.2 Generating Body Plans by Combining Parts

The three sets of part detections; head, torso, and legs, from the previous section, are then combined to generate candidate body plans. Subsets of the three types of components represent a multi-part body hypothesis. According to the study of [12], most occlusions for pedestrian occur over the lower body or the side of a pedestrian, and only a few patterns of these occlusions account for more than 90% of all occlusions. With this in mind, we only use 3 types of body part combinations, corresponding to the example, shown in Figure 4.1; head only (H), head and torso (HT), or head torso and leg (HTL). If nH heads, nT torsos and nL legs are detected, there could be nH + nH X nT + nH X nT X nL different subsets to consider as candidates. However, not all combinations of all parts are needed, since a multi-part body hypothesis needs to comply with the expected spatial geometry.
of a body configuration. For example, a person's head should be above their torso, and the torso needs to be above the legs, all within a reasonable distance of each other. Algorithm 4.1 summarizes steps used to combine the components of parts into body plans with respect to geometric orientation and distance constraints.

For a two-part body plan, each head is paired with the highest confidence torso, if one exists, that is within a maximum distance. For each torso that is paired

**Algorithm 4.1** Combining parts into body plans

- $H \leftarrow$ all head detections
- $T \leftarrow$ all torso detections
- $L \leftarrow$ all leg detections,
- $S_1 \leftarrow H$,  \hspace{1cm} $\triangleright$ $S_1$ is an output set containing one part
- $S_2 \leftarrow \emptyset$,  \hspace{1cm} $\triangleright$ $S_2$ is an output set containing two parts
- $S_3 \leftarrow \emptyset$,  \hspace{1cm} $\triangleright$ $S_3$ is an output set containing three parts

for each $h$ in $S_1$ do
  
  $X \leftarrow \{h, t\} < d_{\text{max}}$
  
  $t' \leftarrow \text{maxunary}(X)$
  
  $S_2 \leftarrow \{h, t'\}$

end for

for each $t'$ in $S_2$ do
  
  $X \leftarrow \{t', l\} < d_{\text{max}}$
  
  $l' \leftarrow \text{maxunary}(X)$
  
  $S_3 \leftarrow \{\{h, t'\}, l'\}$

end for
With a head, we add the highest confidence leg, if one exists, that is within same maximum distance.

### 4.4 Our Approach

We proposed an optimization framework that uses body plans with three body components as detection candidates. Using candidates with different number of body parts allows us to reason over the occlusion. The framework needs an excessive number of detection candidates, such that evidence for an occluded person is included in the detection candidates with high probability. Among these possible candidates, our optimization approach then selects a subset of candidates, depending on the amount of overlap and occlusion, as a final detection result. An overview of the proposed framework is shown in Figure 4.6. The framework mainly consists of two processes; generating body plans as candidates and finding an optimal set of
candidates as a final detection. In the first part, part based detectors are applied to produce a set of detection bounding boxes with associated confidence scores. We modified the part based detector from [19][24] to generate three types of body parts. Setting a low confidence threshold, a large number of detections can be produced. The three body components are shown as red (head), green (torso), and blue (leg) boxes in Figure 4.6. The next step is to find a combination of these parts to generate a set of multi-part body plan hypotheses, as described in the previous section. Then, overlap and occlusion are measured among these candidates and used to form a quadratic objective function for deciding which candidates to keep. Section 4.4.1 and 4.4.2 explain how the pairwise scores for overlap and occlusion are determined. A binary quadratic optimization procedure decides which candidates to keep or discard by searching for an assignment of 0’s and 1’s to body plan candidates that yields a high, ideally the maximum, objective function value. Section 4.4.3 explains how to formulate this detection problem as a Quadratic Unconstrained Binary Optimization (QUBO) problem. The final detection solution is shown at the lower left of Figure 4.6 as body plans (red, green, and blue boxes) and yellow bounding boxes that also include the expected size and location of the missing parts if the body plan does not contain all components.

4.4.1 Overlap Among Candidates

Similar to the pairwise overlap term in Chapter 3, the pairwise overlap here is a penalty term determining the ratio of pixel overlap among two body plan candidates. This overlap penalty prevents selecting too many candidates at the same or closely neighboring location by imposing a penalty for excessive amounts of overlap between two body plans. If there is only a small overlap penalty, it might be
reasonable to keep both body plans. On the other hand, if there is a large overlap, a higher penalty is applied, and it is probably better to remove one body plan as both of them may belong to the same person. The overlap ratio is computed as the intersection area (in pixels) of the two overlapping body plans, divided by the total area (in pixels) of the smaller body plan. Any missing parts are not used in determining the overlap area, only the visible (detected) parts are used. Two examples of computing pixel overlap area, illustrated by gray shaded area, are shown in Figure 4.7.

Figure 4.7: Two examples of a pixel overlap area between two body plans. Left: overlap between a two-part and three-part body plan. Right: overlap between two three-part body plans.

4.4.2 Occlusion Reasoning

When multi-person occlusion occurs, some body parts of the occluded person are partially or totally occluded by another person. The occluder should be in front of the occluded person, when looking from the camera viewpoint. If the ground plane is flat, the location of the occluder’s feet in image coordinates should also be lower than the occluded person. Furthermore, the height of occluder should also be taller as well. This information can help decide if a body plan candidate with
a missing part can be explained as being occluded by another candidate or not. A candidate with missing parts is more likely to be the result of an occlusion when the following occlusion criteria are met.

- The feet location of the occluded person candidate in image coordinates is higher than the potential occluder.
- The height of an occluded person candidate in pixels is shorter than the potential occluder.
- We have the assumption that the occluder should contain an equal or more number of detected parts. For example, a head-only body plan can be occluded only by either a head-only, head-torso or head-torso-leg body plan. Similarly, a head-torso body plan can be occluded only by either a head-torso or head-torso-leg body plan.

For a candidate meeting the above criteria, we determine the amount of pixel overlap ratio as an occlusion pairwise score that measures the overlap ratio between a hypothetically missing body part of a candidate with another body plan candidate. For a candidate not meeting the above criteria, we use the amount of pixel overlap ratio as an occlusion pairwise penalty score. This penalty score penalizes the improper occlusion such as a H-T body plan is occluded by a smaller H-T-L body plan. Figure 4.8 shows an example of overlap between a body plan and a neighboring body plan. The red, green, and blue boxes are a head, torso, and legs respectively. The dashed line of a green or blue box represents a missing part and its expected location. The light shaded area is the amount of occlusion for calculating overlap between missing body parts and a neighboring body plan. In Figure 4.8 (a) and (b), only the head is detected for Person A. Comparing the
Figure 4.8: Occlusion reward ratio between a body plan hypothesis (red, green, and blue boxes) and a neighboring body plan hypothesis. Solid lines represent detected parts and dotted lines represent missing parts. For Person A, (b) and (d) are better candidates than (a) and (c) respectively, as explained in the text.

Overlap of the missing parts (light shaded area), about half of the missing parts in (b) are overlapped while there is only little overlap for (a). This indicates that, for Person A, (b) is a better selection than (a) because the missing parts may be due to the occlusion. In Figure 4.8 (c) and (d), both head and torso are detected for Person A. Comparing the overlap of the missing parts (light shaded area), more than half of the missing parts in (d) are overlapped while there is only little overlap for (c). This indicates that (d) is a better selection than (c) because the missing leg may be due to occlusion. Basically, a candidate, that meets the occlusion criteria and has a missing part that overlaps with another candidate receives an occlusion reward depending on the amount of pixel overlap. This is a reward rather than a penalty because the visible part of the overlapping candidate explains the absence of the missing part of its neighboring candidate.
4.4.3 Problem Formulation

As in Chapter 3, we formulate the problem of selecting a subset of body plans as a Quadratic Unconstrained Binary Optimization (QUBO) problem. Binary optimization is the problem of finding a binary vector \( x = [x_1, x_2, ..., x_n] \) that maximizes an objective function \( f(x) \). Each \( x \) represents a body plan in this case. As before, we use a quadratic objective function:

\[
f(x) = \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} x_i x_j + \sum_{i=1}^{n} c_i x_i \tag{4.1}
\]

where \( x_i \in \{0,1\} \) and \( i = 1, ..., n \). \( x_i \) are the binary variables to be solved for, \( c_i \) are \( n \) unary coefficients, and \( c_{ij} \) are \( O(n^2) \) pairwise coefficients. The goal is to assign 0,1 values to each \( x_i \) in a way that maximizes the objective \( f(x) \), and we combine all the coefficients \( c_i \) and \( c_{i,j} \) into a single square matrix \( Q \) and solve the binary integer maximization

\[
\max_{x \in \mathbb{B}^n} f(x) = \max_{x \in \mathbb{B}^n} x'Qx \tag{4.2}
\]

The key aspect of this optimized detection problem is to decide which multi-part body hypotheses to keep and which to discard. In this framework, we integrate reward (positive values) and penalty (negative values) parameters into the objective function using unary and pairwise terms. Unary terms represent a measurement of each individual candidate while pairwise terms represent measurements between candidates.

- \( u \) is a confidence score from the part based detector. Each body plan is associated with a detection confidence score. This value determines how good each body plan is.
• $b$ is a pairwise overlap among each body plan which is an amount of pixel overlap between the detected parts of the two pairwise body plans. This is a penalty parameter that prevents selecting multiple body plans at the same or nearby locations since they could belong to the same person.

• $o$ is an occlusion reward. This is a parameter that rewards candidates where a missing body part, such as leg, is occluded by visible parts of another body plan. This Pairwise measurement between a multi-part body plan is a measurement that allows reasoning about occlusions.

• $p$ is an occlusion penalty. Unlike overlap penalty that is amount of pixel overlap between detected parts, occlusion penalty is the amount of pixel overlap between the missing part of one body plan (being occluded) and detected part of another body plan (occluder). This penalty is applied if the missing body part is occluded by a body plan with an unreasonable size or location which is opposite to occlusion reward.

The unary term is $u$ and the pairwise terms are $b$, $o$ and $p$. These four parameters form the matrix $Q$ of our objective function as shown in Figure 4.9 where

$$A = u$$

$$B = -b + o - p$$

The rows and columns of $Q$ represent each body plan $[B_1, B_2, ..., B_m]$. It may seem that an alternative formulation would be to use individual body parts (head, torso, and leg) as rows and columns of $Q$ rather than first combining them into multi-part body plans. However, that approach is problematic because the binary vector solution then cannot indicate which parts belong to the same body, and
hence cannot adequately reason about the occlusion overlap. Using body plan hypotheses formed previously also results in a much smaller size of matrix $Q$ than would be the case if all body parts were used directly. In this framework, we use quadratic programming in the same way as described in Chapter 3 to solve eq. 4.2. The quadratic proograming does not directly optimize the binary problem, but instead optimizes a continuous relaxation where variables take values between 0 and 1. A rounding function is then applied to the continuous solution to recover a binary solution.

### 4.4.4 Weight Parameter Learning

For matrix $Q$, different observations are combined as $Q = w_1 u - w_2 b + w_3 o - w_4 p$. To properly incorporate this different information, we will need to specify relative weights for fusing different types of observations into a single pairwise term. We can think of learning weight parameters as finding a set of weights that maximizes a cost function measuring accuracy on a set of training images. These weights can be learned using the pattern search method described in Section 3.2.2. We use Pattern Search to find values for $w_1$, $w_2$, $w_3$ and $w_4$ that maximize the accuracy over six training images per dataset. If the camera viewpoint in one dataset is

![Figure 4.9: Matrix $Q$ for objective function in eq. 4.2](image-url)
Figure 4.10: Graph showing cost function (1-accuracy) during each iteration of the pattern search procedure that solve for relative weights $w_1, w_2, w_3, w_4$ that maximize system detection accuracy on a training set.

Different from another dataset, the learned weights can be slightly different. In the experiment, since the pattern search is finding a minimization, we define the cost function as $(1\text{-accuracy})$. Figure 4.10 shows the cost function, at each iteration of the search, while learning weights for the Terrace dataset. The number of function evaluations at each iteration is different depending on the mesh size and how early the polling stops. For the Terrace training set, we found weight parameters as $w_1 = 0.0781$, $w_2 = 0.8826$, $w_3 = 0.1174$, and $w_4 = 0.9219$ with the number of iterations being 40 and the total number of function evaluations being 260. The best cost function value is 0.19643 which means the accuracy 0.80357 on the training set.

### 4.5 Experimental Results

We evaluated the proposed method using the EPFL multi-camera pedestrian dataset (Terrace sequence) [4], Caviar, and Art Fest datasets. The first video sequence
contains multiple people walking, and there is a large amount of overlap occlusion while the second one contains multiple people walking with small overlap, seen from a further viewpoint. The third one contains a medium density crowd and has a higher number of people than the other two datasets. For baseline methods, we compare our proposed framework with Grammar Models [25][24] and our previous Optimized Detection [61] approach from Chapter 3 that uses full body detection candidates.

### 4.5.1 Results

Table 4.1 shows detection performance of the 3 methods with percentages of true positive (correct match), false negative (missed detection), and false positive detections. The detector accuracy and precision are shown in Table 4.2. For Terrace sequence, our method has slightly fewer true positives compared to the Optimized Detection in Chapter 3. However, the accuracy and precision evaluation show that
Figure 4.11: Qualitative evaluation shown as ROC curve. (top) Terrace Dataset (middle) Caviar Dataset (bottom) Art Fest Dataset.
Figure 4.12: Our results are shown in (a) and (b) where (a) shows detected body parts and (b) shows final bounding boxes. (c) Optimized detection (Chapter 3) (d) Grammar model [25]
Figure 4.13: Examples of some incorrect detection results using our method such as placing a body plan on an incorrect part of the body or a head floating near another candidate.

The performance is better. As for Caviar and Art Fest dataset, our method performs better than the other two baseline methods. The number of true positives is higher, as well as the accuracy and precision scores. The qualitative evaluation as a ROC curve is shown in Figure 4.11. The ROC curve plots false positive rate against true positive rate. Sample detection result images are shown in Figure 4.12. Our results are shown in (a) and (b) where (a) shows body plans with the detected body parts and (b) shows the final bounding boxes that also include the expected size and location of the missing parts if the body plan does not contain all components. Figure 4.13 shows some examples of incorrect detection results from our method.
4.5.2 Quality of the Solution

Since directly solving for a binary solution is not easy. We, instead, solve the relaxed continuous version of the problem using quadratic programming. The binary result after applying a rounding function seems to yield a good solution even without directly solving for it. We examined over 8,000 values of the solution variables to determine the quality of our relaxed solutions and find that the actual values of the solutions prior to rounding are either very close to zero or one as shown in Figure 4.14. For these solution values, they are at most $3.8913 \times 10^{-13}$.

![Figure 4.14: Actual solution values solved by quadratic programming on the relaxed continuous problem. In all cases examined, the continuous solution vectors were indistinguishable from a binary solution.](image)
from one and at most $5.8620 \times 10^{-14}$ from zero.

While not providing conclusive evidence, the fact that solution vectors to the relaxed problem are very nearly binary even before rounding suggests that the quadratic programming relaxation is finding a good approximation to the binary solution of the original problem. It further suggests that the actual method used to binarize the relaxed solution (we use rounding) does not affect the solution at all.

4.6 Conclusion

We have proposed a framework for improving occluded pedestrian detection performance using an optimized detection framework from Chapter 3 with multi-part body plans. The multi-part body plans allow us to reason directly about the occlusions among candidates, which helps decide whether a candidate with missing body parts is likely due to being occluded by another candidate. Our qualitative results show that the use of multi-part body plans within an optimization-based detection framework yields a performance improvement over state-of-the-art approaches using only part detectors or our earlier approach using optimized detection with only full body detection candidates.
Chapter 5

Optimized Detection Coupled with Data Association

5.1 Introduction

Object tracking, especially pedestrian tracking, has been of much interest in the computer vision research community. The tracking of individuals in cluttered scenes is an important aspect of a video surveillance system. This type of system requires each pedestrian to be tracked to generate a path or trajectory across multiple frames. Likewise, multiple individuals need to be correctly matched to maintain identity of the same individual across a sequence of frames.

Occlusion often arises in multiple target tracking. When there is a group of pedestrians walking in the scene, it is quite common that people walk past one another and some parts of their bodies are occluded. When occlusion occurs, the occluded person may not be detected, which may result in a fragmentation of trajectories or a mismatch. In some cases, even without occlusion, the person may be hard to detect due to color of clothing or body pose, resulting in the same way
in the fragmentation of trajectories. A robust tracking system needs to be able to handle these missing detections and the occlusions issue.

Pedestrian detection and pedestrian tracking are both challenging problem by themselves. Several approaches usually divide the processes into finding detections first and then solving tracking from the detection results by matching set of detections in consecutive frames across sequences to find trajectories for all the targets. The weakness of the detect-then-track approach is that tracking performance also depends on the reliability of the detection output. If the detection output contains a miss or false positive detection, the tracking framework needs to be able to cope with these issues.

Frequently, a partial occluded person in a particular frame is not occluded in a subsequent frame seen a short time later. The amount of occlusion is usually incrementally increased or decreased in consecutive frames. An example in Figure 5.1 shows a person (wearing black in the middle) who is not occluded in the first frame but is later occluded in the following frames with some visible body parts. This information motivates us to combine detection with data association as a single problem. We do not only find object matches between two frames but also solve for detection results in order to improve the detection results for occluded objects while improving matching results for data association as well.

Figure 5.1: Example frames showing an unoccluded person in the first frame being occluded in the next two frames.
Instead of using the detect-then-track approach, we propose a framework that explicitly reasons about multi-frame detections coupled together with data association. This coupling framework can improve both detection and tracking performance. We apply our optimized detection framework to two consecutive frames and formulate the problem as selecting tracklet candidates instead of detection candidates. Nevertheless, the tracklet candidates also represent selected detection candidates from both frames.

**Contributions in This Chapter**

- We propose a framework that incorporates detection together with data association using a similar optimization framework as Chapter 4. The coupling framework solves for a solution that represents both detection and association of people across two consecutive frames.

- We reformulate the non-bipartite graph matching problem into Quadratic Constrained Binary Optimization to solve for a subset of matchings that maximize the objective function.

### 5.2 Our Approach

We propose to extend our optimized detection framework presented in Chapter 3 and Chapter 4 to be applicable to the data association problem of matching multiple objects across two frames. This coupling framework incorporates the detection process with the data association process. We reformulate the previous framework of quadratic unconstrained binary optimization problem to quadratic constrained binary optimization. Our aim is to solve for a set of detections in two frames while simultaneously matching those detections across the two frames.
Figure 5.2: (left) A set of detection candidates shown as colored bounding boxes. (right) Graphical representation of the set of detections in the left image. The color of the node indicates which candidate in the left it belongs to.

Figure 5.2 shows a graphical representation of the optimized detection framework in Chapter 4. An example set of detection candidates is shown by colored bounding boxes in the left figure. The detection candidates are converted to nodes as shown in the graphical representation on the right. Each node, $X_1$ to $X_5$, represents a detection candidate along with an associated detection confidence score. The undirected edges depict pairwise scores, which represent overlap and occlusion, as explained in Chapter 4.4.1 and 4.4.2.

In this framework, we acquire sets of detection candidates from two frames as shown in Figure 5.3. The goal of our problem is to find a subset of matching pairs from among these candidates. The resulting matching pairs represent both matching and the candidates to select in the two frames. So, instead of selecting full-body detection candidates (Chapter 3) or body plans (Chapter 4) as binary variables, in this chapter we select a set of matching pairs (i.e. tracklets) from a set of possible tracklet candidates.

The following sections present more detail of this problem formulation. Section 5.2.1 describes how to generate tracklet candidates, whereas Sections 5.2.2 and 5.2.3 explains how the optimization problem is set up. Unlike the traditional
two-frame matching problem which can be represented as a bipartite graph and solved optimally by efficient algorithm, our formulation has pairwise costs representing occlusion/overlap of nodes in the same frame, and therefore our matching problem is non-bipartite. We reformulate the non-bipartite problem into quadratic constrained binary optimization problem. The reformulation is explained in Section 5.2.3.
5.2.1 Generating Tracklet Candidates

In this work, a tracklet represents a matching of an object between two consecutive frames, which can further be used to indicate the displacement of that object along a trajectory in the frame sequence. We generate tracklets as our possible candidates for matching objects between the two frames and associate a binary variable with each tracklet in our optimization framework.

Given a two sets of candidates in two frames, Figure 5.4 shows examples of
tracklets as blue undirected edges connecting pairs of detection candidates from two adjacent frames. We want to find a subset of tracklets that form a matching between objects in the two frames. The decision to keep or discard each tracklet candidate is represented as a binary variable where 0 means to discard and 1 means to keep. Since each tracklet is associated with a pair of detection candidates in both frames, solving for a set of tracklets also yields a subset of detection candidates from both frames.

Generating possible sets of matches can produce a large number of matching
candidates. For example, if there are 100 detection candidates in frame 1 and 150 detection candidates in frame 2, there can be as many as 15,000 possible tracklets. However, most of these tracklets are not likely due to constraints on distance, size similarity, etc. We only want to generate a likely set of tracklet candidates that meets specific criteria. For example, detection $X_1$ in Figure 5.4 can be matched to $Y_1$, $Y_2$ or $Y_3$ since they are within a reasonable distance. Candidate $Y_4$, $Y_5$, and $Y_6$ are too far for $X_1$ to have moved to those locations in the time elapsed between frames.

Furthermore, we do not want to assume that all the objects need to be matched. For example, an object may not have a match because the object in the next frame may be totally occluded or the object is otherwise not well detected (the object is not detected as a candidate). If the confidence of a detection in one frame is strong enough, that detection candidate should be kept but left as a no-match, meaning it is not matched to any candidates in the other frame. To resolve this matter, we add a dummy node to each frame to represent a virtual match with any selected candidate that becomes a no-match. Figure 5.5 shows adding dummy nodes, $X_0$ and $Y_0$. Undirected edges connect dummy nodes to all detections in the other frame.

To summarize, the criteria for generating tracklets in our experiment are:

- Both dummy nodes connect to all nodes in another frame, to represent the possibility of a strong candidate in one frame that has no match in the other frame.

- We compare the distance between the center location of each detection candidates (center of the full body bounding box) and generate a tracklet if the distance is less than a specified value constraining the distance traveled
between frames. For the Caviar dataset, for example, we use 40 pixels as a cut-off value.

5.2.2 Problem Formulation

We now formulate the problem of selecting tracklets as non-bipartite graph matching, as shown in Figure 5.5. There are two sets of nodes, representing detection candidates, from two consecutive frames. Within each set, a node may be connected with another node in the same frame by an undirected edge, associated with a pairwise score, if there is a pairwise relation such as overlap or occlusion between them. Undirected edges connecting nodes between the two sets represent a tracklet or fragment of the trajectory of a matched object. We associate a binary variable with each tracklet edge, allowing a match to be turned on or off, subject to the objective function.

The tracklet edges are also associated with unary scores. Since selecting a tracklet means selecting one candidate in each frame, the unary score of each node in each frame (based for example on detector confidence) is combined as a unary score for the tracklet. This positive unary score determines how good the pair of detection candidates is. Other types of information that can be used to determine whether a match is good includes distance between the pair of matches candidates locations in the image. A good match should be nearby since an object cannot move very far between frames. It is very likely to match a candidate to the closest one in the next frame. Other information such as similarity of heights of the two objects also suggests how likely the two objects are the same. We use distance and height difference as penalty unary scores to discourage selecting pairs that are less likely to be good matches.
Finally, there is an additional constraint that did not arise in either of our two earlier problem formulations. Matching problems have a one-to-one matching constraint such that an object in either frame can match at most one object in the other frame. That is, an object in the first frame can match to only one object or to the dummy node (no match) in the second frame. And vice versa, an object in the second frame can be matched to at most one object in the first frame. A no-match is indicated by matching to the dummy node and unlike “real” nodes, dummy nodes are allowed to participate in multiple matches.

In our example, node $X_1$ can either be matched to a dummy node $Y_0$ as no-object matching, or node $Y_1$, $Y_2$, $Y_3$ as a matching to detection $Y_1$, $Y_2$, $Y_3$ respectively. A feasible matching solution can only contain either one of these four edges. These constraints can be represented as a set of linear constraints. Below is an example of the matching constraints for nodes $X_1$, $X_2$, and $X_3$ in Figure 5.5.

\[
e_1 + e_2 + e_3 + e_4 \leq 1
\]
\[
e_6 + e_7 + e_8 + e_9 + e_{10} \leq 1
\]

### 5.2.3 Non-Bipartite Graph Reformulation

In order to solve the problem of selecting tracklets in the non-bipartite graph as a quadratic constrained binary optimization, we reformulate the graph by creating a new node from each of the tracklet edges, as depicted by the square nodes in Figure 5.6. Each square node represents a matching edge (tracklet) in the original graph, or equivalently, a pair of matching detection candidates. Figure 5.6 illustrates the transformation of a non-bipartite matching graph, on the left, to a new graph on the right. The reformulated graph represents a quadratic constrained
binary optimization problem. During solution of this problem, the linear matching constraints shown at the bottom left must also be enforced.

Each square node in the transformed graph is also associated with two detection nodes in the original graph on the left. Since these two nodes are combined, their pairwise cost edges need to be propagated to the new graph. The color of edges in the figure indicates where the pairwise edge costs are propagated to. Pairwise costs are combined if they are both associated with the same pair of nodes in the new graph. For example, if we select $e_1$ and $e_3$ as our solution, $X_1$ and $X_2$ detections are selected in the first frame while $Y_1$ and $Y_2$ are selected in the second frame. The pairwise cost of the red edge is applied to detections in the first frame, and the pairwise cost of the blue edge is applied to detections in the second frame. So, for the new graph, if we select $e_1$ and $e_3$, their pairwise cost is the combined value from the red and blue edges. If the red edge has a pairwise cost of 0.1 and the blue edge has a pairwise cost of 0.3 in the original graph, the pairwise cost between $e_1$ and $e_3$ in the transformed graph is 0.4.

Nodes in the transformed graph (square nodes) are now binary variables we want to solve for. These nodes have associated unary scores as explained in Section 5.2.2 and there are pairwise scores between pairs of nodes as explained above. The problem in the reformulated graph becomes a Quadratic Constrained Binary Optimization problem, with the constraints being the linear matching constraints. The quadratic objective function is defined as

$$\max_{x \in \mathbb{B}^n} f(x) = \max_{x \in \mathbb{B}^n} x^TQx \quad \text{subject to linear matching constraints} \quad (5.1)$$

where $x \in \{0, 1\}$ are the square nodes and square matrix $Q$ contains unary coeffi-
Figure 5.6: An example of to non-bipartite matching graph reformulation. (left) A non-bipartite matching graph with three candidates, $X_1$, $X_2$, and $X_3$, in the first frame and two candidates in the second frame, $Y_1$ and $Y_2$. Its linear constraints are shown on the bottom left. (right) The reformulated graph is represented as a quadratic constrained binary optimization problem. The square nodes represent undirected edges in the left graphical model. Color edges depict the regenerated pairwise edges.

\[
\begin{align*}
\text{s.t.} & \quad e_i \in \{0,1\} \\
& \quad e_1 + e_7 \leq 1 \\
& \quad e_2 + e_3 + e_8 \leq 1 \\
& \quad e_4 + e_9 \leq 1 \\
& \quad e_1 + e_2 + e_5 \leq 1 \\
& \quad e_3 + e_4 + e_6 \leq 1 
\end{align*}
\]
cients and pairwise coefficients.

\[ Q = U + P \]  \hspace{1cm} (5.2)

where

\[ U = w_1 u_1 - w_2 u_2 - w_3 u_3 \]

\[ P = w_4 p_1 - w_5 p_2 - w_6 p_3 \]

\( U \) is a diagonal unary score matrix containing unary confidence and unary penalties where \( u_1 \) is a detection confidence score of the two matching candidates, \( u_2 \) is the distance between the two matching candidates, and \( u_3 \) is the difference in height of the two matching candidates. \( P \) is a pairwise score matrix containing both pairwise rewards and penalties, where \( p_1 \) is an overlap penalty, \( p_2 \) is an occlusion reward, and \( p_3 \) is an occlusion penalty of the pair of candidates.

To solve for a quadratic constrained binary optimization problem, we also tried IBM ILOG CPLEX optimizer toolbox which is a mathematical programming solver for linear programming, mixed integer programming, and quadratic programming. CPLEX uses branch and cut algorithm when solving a binary optimization problem. However, computational time is extremely slow. It took several hours for a problem with a size of about 200 tracklet candidates. So, similar to previous framework, we solve the quadratic constrained binary optimization problem using quadratic programming by relaxing the objective function to be continuous and then round the solution to be binary. However, the linear constraints on the solution needs to be taken into account when converting the solutions into binary.
5.3 Experimental Results

We evaluated the proposed method using Caviar dataset that has person id available in the ground truth for matching evaluation. Table 5.1 shows quantitative result for matching along with detection in the two consecutive frames. The result shows that the proposed method can perform data association while also maintain good detection results. The matching false positive indicates an id swap where a person is incorrectly matched to a different person.

The detection result images are shown in Figure 5.7 and 5.8. Detection bounding boxes are shown as full body (yellow) and body parts (red, green, and blue). The matching results are depicted as id number on the bounding box. For example, bounding box 1 is matched to bounding box 1 and bounding box 2 is matched to bounding box 2 in the subsequent frame, shown on the right. Figure 5.7 (a) shows a case where detections, number 4 and 5, are kept but being no-match to the other frame. Figure 5.7 (b) shows an occlusion case where two people are walking past each other with person number 2 only has a head visible. And yet, the method is able to handle such occlusion and correctly detects and matches the two persons. In Figure 5.8 (b), there are person number 1, 2, 3, 4, and 5 in the first frame while there are person number 1, 2, 3, 4, and 6 in the second frame. In this case, person 5 and 6 are both detected but are not matched to each other. Since person 6 only contains head, the height has to be estimated based on the size of head. Too much different in height may cause the detection candidate to be unmatched.
<table>
<thead>
<tr>
<th></th>
<th>TP</th>
<th>FN</th>
<th>FP</th>
<th>Accuracy</th>
<th>Precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>Detections in Frame 1</td>
<td>0.56</td>
<td>0.44</td>
<td>0.043</td>
<td>54.1</td>
<td>93.0</td>
</tr>
<tr>
<td>Detections in Frame 2</td>
<td>0.55</td>
<td>0.45</td>
<td>0.040</td>
<td>53.2</td>
<td>93.3</td>
</tr>
<tr>
<td>Matching between frames</td>
<td>0.51</td>
<td>0.49</td>
<td>0.043</td>
<td>48.7</td>
<td>92.3</td>
</tr>
</tbody>
</table>

Table 5.1: Quantitative result for simultaneous detection and matching across pairs of frames. Table shows percentages of true positive (correct), false negative (miss), false positive, accuracy, and precision.

### 5.4 Summary

This chapter has proposed an extension to show that the optimized detection framework can be applied to two consecutive frames to find detection results along with data association. Detection and matching candidates are being solved together, unlike other approaches that perform detection first and then associate those detections to find matches. Detection and matching across two frames can be represented as a non-bipartite matching graph, which we reformulate into a quadratic constrained binary optimization problem. The extension is currently limited to data association between two consecutive frames. However, to fully develop this approach into a tracking framework, a future work needs to be done to combine the two-frame tracklets into longer trajectories across the whole sequence.
Figure 5.7: (a) Frame 455. (b) Frame 460. The subsequent frame of (a) and (b) are shown on the right. The numbers on bounding boxes indicate matching id. Detection bounding boxes are shown as full body (yellow) and body parts (red, green, and blue).
(c) Result: 5 correct matches, 5 detections in frame 1 and 5 detections in frame 2

(d) Result: 4 correct matches, 5 detections in frame 1 and 5 detections in frame 2. Detection id 5 and 6 are not matched.

Figure 5.8: (a) Frame 480. (b) Frame 505. The subsequent frame of (a) and (b) are shown on the right. The numbers on bounding boxes indicate matching id. Detection bounding boxes are shown as full body (yellow) and body parts (red, green, and blue).
Chapter 6

Conclusion and Future Work

6.1 Conclusion

This thesis presents a binary quadratic optimization framework for multi-pedestrian detections. The proposed framework is intended to improve detections on a group of pedestrians with partial occlusions. We propose to formulate the pedestrian detection process as a quadratic binary optimization. This formulation reasons in about overlapping object detections as a pairwise measurement, to optimize the tradeoff between detection confidences and amounts of overlap, allowing a more principled approach to select the final set of detections then non-maximum suppression. The core concept of optimized detection is to generate a large set of possible detection candidates and then use the optimization framework to select the best set of candidates. The candidate can be a full body detector or a body plan containing several body parts. It is also applicable to use tracklets formed from two-frame matches as candidates. Figure 6.1 shows the core concept of the proposed optimized detection framework where different types of candidates can be used. The optimized detection framework, described in Chapter 3, shows that
quadratic unconstrained binary optimization for reasoning about overlapping detections can improve the performance of a pedestrian detection system, especially when there are multiple occluded persons. The optimization is formulated as a quadratic objective function that contains both unary scores measuring the quality of an individual detection, and pairwise scores measuring the joint compatibility of pairs of overlapping detections. Although this is an NP-hard problem to solve exactly, efficient approximate methods are applicable that yield high quality solutions on large problem sizes. For an occluded person, a full body detector often produces a low detection confidence. To overcome this limitation, we adapt the
framework, as presented in Chapter 4, to body plans that consist of multiple body parts, in order to gain a better detection confidence for each unoccluded part. A body plan is a representation of a full body with missing parts. Generally speaking, using body plans also allows us to perform more detailed occlusion reasoning by trying to understand whether a body part is undetected because of occlusion with other body plans or not. To help with the occlusion reasoning and candidate selections, reward and penalty values are applied to each detection. An occlusion reward is given to an occluded part if the occlusion can be explained by a closer, visible person. However, when the location and size of candidates suggest that occlusion is unreasonable, an occlusion penalty is given instead. The proposed framework is intended to handle detections with occlusion. However, it is still limited to partial occlusion. When a full occlusion occurs, detectors are not able to generate a possible candidate for such person. As long as the correct detection is not included in the set of generated candidates, the correct solution for that configuration of people cannot be acquired. Total occlusion is quite difficult to handle on a single frame basis. Building upon the proposed optimized detection framework, we extend the approach to handle simultaneous detection and data association between two consecutive frames in Chapter 5. Our approach is to combine detection and tracklet candidates together before solving for the final set, unlike other approaches that perform detection first and then tracking. To apply the optimized detection concept to data association, we use tracklets as candidates instead of detection responses. The problem is represented as a non bipartite graph with matching constraints, which we propose to reformulate as a quadratic constrained binary optimization problem with linear constraints. Our framework is able to fuse several different information sources such as amount of pixel overlap, detection confidence score, or distance among candidates. Directly adding these
values together would not yield a good solution. In such case, it is desired to learn appropriate weighting values. We propose a search-based learning approach that finds a good set of weight values without knowing the gradient optimization of the function, which seems to maximize the accuracy of the detection system on training images. In summary, we propose an optimized detection framework for detecting multiple partially occluded pedestrians. The framework offers a smarter way to select a final set of detection candidates instead of applying a hard threshold or non-maximum suppression. The framework is strengthened by the capability to reason about occlusion among pairs of detection candidates when deciding which candidates to keep. The framework can also be applied to a coupled framework for simultaneous data association and detection.

6.2 Related Publications


- The work presented in Chapters 4 and 5 will be submitted to BMVC 2014 (British Machine Vision Conference) on May 2, 2014.

6.3 Future Works

The use of a three-part body representation allows a better understanding of occlusion than over a full body method. Future work can further apply the framework to a representation with more than three body parts. Using more number parts
would allow a more comprehensive reasoning about occlusions. In addition to allowing more precise localization of the occlusion, it can also enable more nuanced reasoning about the pattern of occlusion and where the occlusion might occur. For example, if a torso is slightly overlapped with another body plan on the side of the torso, the leg should still be visible, so the selected candidate is likely to contain a leg part. On the other hand, if a torso is slightly overlapped at the bottom of the torso, the leg should be missing because occlusion is occurring at the lower part of the body. Using the optimized detection framework to solve two-frame data association should be further extended for multi-frame tracking. One approach is to combine individual tracklets to create a full trajectory path of an object. So, a set of tracklets in the current frame pair needs to be matched to the set of tracklets in subsequent frame pairs to generate the whole sequence of linked tracklets that represent the trajectory for each object. For the data association framework, we only use the distance and difference in height among candidates to measure matching likelihood. However, other useful information can be incorporated to improve the matching. A good source of information that can be used to indicate a proper match is clothing color. Two candidates with similar color of clothing should be a better match than ones with different colors. The location or path from a previous frame can also be used to roughly predict the possible location in the next frame where matched candidates should appear.
Bibliography


Vita
Sitapa Rujikietgumjorn

Education

**Ph.D. in Computer Science and Engineering**
*The Pennsylvania State University*
Advisor: Assoc. Prof. Robert T. Collins
Dissertation: “Quadratic Binary Optimization for Pedestrian Detection in Crowded Scenes”

**M.S. in Computer Engineering**
*The University of Tennessee*

**B.S. in Computer Engineering**
*Khon Kaen University, Thailand*

Professional Experiences

Research Assistant, Laboratory for Perception, Action and Cognition, *The Pennsylvania State University*
Aug 2012 - May 2014

Teaching Assistant, The Pennsylvania State University
- Programming for Engineers with MATLAB, 2010-2012
- Fundamentals of Computer Vision, 2009

Master Student, supervised by Prof. Mongi Abidi
*Imaging, Robotics and Intelligent Systems Laboratory, The University of Tennessee*
Jan 2007 - Aug 2008

Undergraduate Exchange Student, supervised by Prof. Mohammad Alsharif
*Signal Processing Laboratory, University of Ryukyu, Japan*
Oct 2004 - Mar 2005

Awards and Honors

- Royal Thai Government Scholarship for Master and Doctoral Degrees, 2006
- AIEJ scholarship from the Japanese Government to study a non-degree exchange program at University of Ryukyu, Okinawa, Japan, 2004
- Honorable award on “Loan Calculation Service” from the Sixth National Software Contest (NSC) in undergraduate level, Thailand, 2004
- Undergraduate senior project on “Developing Motor Control System for Foam Cutting Machine” was supported by the Thailand Research Fund as a part of IPUS (Industrial Projects for Undergraduate Students), 2003