ISTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

REAL TIME SKELETONIZATION ON FPGA WITH A HAND TRACKING APPLICATION

M.Sc. THESIS

Melike ATAY

Department of Electronics and Communications Engineering

Electronics Engineering Programme

JANUARY 2014

ISTANBUL TECHNICAL UNIVERSITY ★ INSTITUTE OF SCIENCE AND TECHNOLOGY

REAL TIME SKELETONIZATION ON FPGA WITH A HAND TRACKING APPLICATION

M.Sc. THESIS

Melike ATAY
(504101230)

Department of Electronics and Communications Engineering

Electronics Engineering Programme

Thesis Advisor: Assoc. Prof. Dr. Müştak E. YALÇIN

JANUARY 2014

İSTANBUL TEKNİK ÜNİVERSİTESİ ★ FEN BİLİMLERİ ENSTİTÜSÜ

ALANDA PROGRAMLANABİLİR KAPI DİZİLERİ ÜZERİNDE GERÇEK ZAMANLI İSKELET BULMA VE EL TAKİP UYGULAMASI

YÜKSEK LİSANS TEZİ

Melike ATAY
(504101230)

Elektronik ve Haberleşme Mühendisliği Anabilim Dalı

Elektronik Mühendisliği Programı

Tez Danışmanı: Doç. Dr. Müştak E. YALÇIN

OCAK 2014

Melike ATAY, a M.Sc. student of ITU Institute of Science and Technology 504101230 successfully defended the thesis entitled **"REAL TIME SKELETONIZATION ON FPGA WITH A HAND TRACKING APPLICATION"**, which she prepared after fulfilling the requirements specified in the associated legislations, before the jury whose signatures are below.

Thesis Advisor :	**Assoc. Prof. Dr. Müştak E. YALÇIN** Istanbul Technical University	....
Jury Members :	**Assist. Prof. Dr. İlker BAYRAM** Istanbul Technical University	....
	Assist. Prof. Dr. Nerhun YILDIZ Yıldız Technical University	....
		....

Date of Submission : 13 December 2013
Date of Defense : 6 January 2014

To my family,

FOREWORD

I would like to express my gratitude to my thesis supervisor Müştak Erhan Yalçın for his invaluable guidance, support and encouragement throughout my thesis. I would like to thank to all my colleagues in TUBITAK UEKAE for their support and strong friendship. My deepest gratitude goes to my family for their love and support throughout my life; this thesis is simply impossible without them. Last but not least I would like to thank to Harun Karabalkan for his love, helpful comments and encouragement.

January 2014

Melike ATAY
Electronics Engineer

TABLE OF CONTENTS

Page

FOREWORD ... **ix**
TABLE OF CONTENTS ... **xi**
ABBREVIATIONS ... **xiii**
LIST OF TABLES ... **xv**
LIST OF FIGURES ... **xvii**
SUMMARY ... **xix**
ÖZET ... **xxi**
1. INTRODUCTION ... **1**
1.1 Motivation ... 1
1.2 Contribution ... 4
1.3 Organisation of Thesis ... 4
2. SKELETONS IN DIGITAL IMAGE PROCESSING ... **5**
2.1 Definitions and Properties of Medial Skeleton ... 5
2.1.1 Maximally-inscribed balls ... 6
2.1.2 Grassfire analogy ... 6
2.2 Skeleton Extraction Methods ... 7
2.2.1 Distance transformation ... 8
2.2.2 Voronoi diagram ... 11
2.2.3 Thinning techniques ... 11
2.2.4 Mathematical morphology ... 12
2.2.5 Comparison of skeleton extraction methods ... 14
2.3 Classification Of Skeleton Points ... 15
2.4 Skeleton Pruning ... 15
3. REAL TIME SKELETONIZATION ALGORITHM AND SIMULATIONS **17**
3.1 Skeleton Extraction Algorithm ... 18
3.1.1 Ridge point detection algorithm ... 19
3.1.2 Extended ridge point detection algorithm ... 23
3.2 Simulation of Skeleton Extraction Algorithm ... 26
3.3 Detection and Classification of Skeleton Points for Hand Tracking ... 33
4. IMPLEMENTATION OF REAL TIME SKELETONIZATION SYSTEM . **37**
4.1 Pipelined Skeleton Extraction Architecture on FPGA ... 38
4.1.1 Camera and DDR2 control ... 38
4.1.2 Preprocessing ... 40
4.1.3 Object detection and morphological cleaning module ... 40
4.1.4 Distance transform calculation ... 41

4.1.5 Ridge point detection module 43
4.1.6 Linking and feature extraction module 44
4.1.7 Hand tracking module 46
4.1.8 DVI driver module 46
4.2 Experimental Results 48
5. CONCLUSIONS AND RECOMMENDATIONS 51
5.1 Future Work 52
REFERENCES 53
CURRICULUM VITAE 57

ABBREVIATIONS

FPGA	**:** Field Programmable Gate Array
GPU	**:** General Processing Unit
MAT	**:** Medial Axis Transform
SE	**:** Structuring Element
DT	**:** Distance Transform

LIST OF TABLES

Page

Table 2.1 : Comparison of Skeleton Extraction Methods. 15
Table 3.1 : Performance Measurement of Images.. 31
Table 3.2 : Performance Measurement of Rotated Images. 31
Table 3.3 : Performance Measurement of the Algorithm in [1]. 31
Table 4.1 : FPGA Resource Usage... 48
Table 4.2 : Total Latency of the System. 48

LIST OF FIGURES

Page

Figure 2.1 : Binary Image of Letter T and its Skeleton. 5
Figure 2.2 : Skeleton of a Rectangle via Maximally-Inscribed Balls [2]. 6
Figure 2.3 : Wavefronts generated by a two point Grassfire Excitation [3]. 7
Figure 2.4 : Masks of Different Metrics (a) Input (b) Cityblock Distance (c) Chessboard Distance (d) Euclidean Distance (e) Chamfer 5-7-11 (f) Chamfer 3-4. 9
Figure 2.5 : Severel Types of Distance (a) Euclidean (b) CityBlock (c) Chessboard (d) Hexagonal (e) Chamfer 3-4 (f) Chamfer 5-7-11. 10
Figure 2.6 : Ridge Points of the Distance Transform. 10
Figure 2.7 : Voronoi Skeleton of a Rectangle (a) Generating (sample) Points (b) Approximating Skeleton [2]. 11
Figure 2.8 : Thinning Process. 12
Figure 2.9 : Samples of Structuring Elements. 13
Figure 2.10 : Opening Process. 14
Figure 2.11 : Branch, Normal and End Points of Letter *T*. 16
Figure 2.12 : Skeleton with Unwanted Branches. 16
Figure 3.1 : Scanning the Rectangle for Four Orientations (0, 45, 90, 135 Degrees). 20
Figure 3.2 : Ridge Points on Distance Transform Image (a) Distance Transform Image (b) Result of y axis Scan (c) Result of x axis Scan (d) Output Image of the Ridge Point Detection Algorithm 22
Figure 3.3 : Test Results : (a), (c), (e) Input Binary Images (b), (d), (f) Skeleton Outputs with Ridge Point Detection Algorithm. 24
Figure 3.4 : Test Results of Extended Algorithm: (a) Set of Ridge Points of a Hand (b) Skeleton Output of a Hand after Linking Process (c) Set of Ridge Points of a Horse (d) Skeleton Output of a Horse after Linking Process. 27
Figure 3.5 : Test Results of Extended Algorithm(cont.): (e) Set of Ridge Points of a Man (f) Skeleton Output of a Man after Linking Process (g) Set of Ridge Points of a Finger (h) Skeleton Output of a Finger after Linking Process. 28
Figure 3.6 : Possible Triangles on the Image 28
Figure 3.7 : Connectivity Numbers 29
Figure 3.8 : Psuedo Code to Calculate Connectivity Number 29

Figure 3.9 : Test Results of Rotated Images : (a) Skeleton Output of a hand with a rotation of 20 Degrees (b) Skeleton Output of a horse with a rotation of 40 Degrees (f) Skeleton Output of a Man with a rotation of 90 Degrees (h) Skeleton Output of a Finger with a rotation 5 Degrees. 32
Figure 3.10 : Filter Mask 33
Figure 3.11 : Possible End Points [4] 34
Figure 3.12 : Possible Branch Points [4] 34
Figure 3.13 : Branchpoints and Endpoints : a)Branch Points and End Points of a Hand b)Branch Points and End Points of a Horse 35
Figure 4.1 : Architecture of FPGA-based Skeletonization System. 39
Figure 4.2 : Morphological Cleaning Block Diagram. 41
Figure 4.3 : Distance Transform Forward and Backward Mask: (a) Forward Mask (b) Backward Mask. 42
Figure 4.4 : Distance Transform Backward and Forward Propagation Circuit. 42
Figure 4.5 : Circuit Implementation of Ridge Point Detection Algorithm. 44
Figure 4.6 : Linking Patterns. 45
Figure 4.7 : Circuit of 2D Convolution. 46
Figure 4.8 : Circuit Implementation of Hand Tracking Module. 47
Figure 4.9 : Timing Diagram of Pipelined Architecture. 47
Figure 4.10 : Full System with FPGA Board and Camera Module. 49
Figure 4.11 : Some Illustrations of the Real Time System. 49

REAL TIME SKELETONIZATION ON FPGA WITH A HAND TRACKING APPLICATION

SUMMARY

People benefit from various systems based on image processing technology throughout their daily routine. Main functions of such systems include object detection, matching, tracking, etc. Skeletonization forms the backbone of many tracking and matching applications.

Skeleton of a shape was first described by Blum in 1967. According to Blum, the skeleton of a 2D shape is the loci of points equidistant from the contour. Skeletonization converts most of the original foreground pixels to background pixels while preserving the skeletal residue on the binary image by eliminating the redundant part. Thus, the shape is represented with small amount of data and analysis of the shape requires less time and resource.

In literature, variety of methods have been developed to calculate the skeletal residue of a shape. Skeletal residue calculation must produce a perfect skeleton to improve the performance of systems which rely on skeletonization. Perfect skeleton should preserve geometrical properties of the shape and it should be one pixel thick. Today, acquiring a perfect skeleton is still a challenging task.

Most of the matching and tracking applications require real time processing capabilities. Nowadays, field programmable gate arrays (FPGA) and graphics processing units (GPU) are usually preferred to meet the real time processing requirements. To date, many implementation methodologies have been developed for image skeletonization on FPGAs. All the implementation methods aim to reduce the resource utilization and power consumption of the systems while increasing the number of processed frames per second.

Due to the necessity of perfect skeleton extraction and real time requirements of these systems, this thesis pursues a real time implementation of skeletonization to be used in many tracking and matching applications. Skeletonization algorithms are usually computationally complex to produce the perfect skeleton. Therefore, most of them are not suitable for real time VLSI implementation. In this work, an extension is suggested to a widely used simple but an efficient skeletonization algorithm. This extension reduces the effect of the boundary noise on the skeleton and produces a skeleton close to a perfect skeleton.

A fully pipelined architecture is proposed to implement the extended skeleton extraction algorithm on FPGA. A complete system is developed on Digilent Atlys board with Spartan-6 FPGA to test the proposed skeletonization architecture. The frames are captured from CMOS image sensor with an integrated advanced camera system MT9D112. The performance is observed on a hand tracking application.

Performance of the novel algorithm is evaluated according to the widely acknowledged performance measures for skeletonization research. Resource utilization and timing performance of the FPGA implementation are investigated for comparison with similar systems in literature.

ALANDA PROGRAMLANABİLİR KAPI DİZİLERİ ÜZERİNDE GERÇEK ZAMANLI İSKELET BULMA VE EL TAKİP UYGULAMASI

ÖZET

Günlük yaşamda trafik görüntüleme sistemlerinden medikal uygulamalara kadar çok geniş bir alanda görüntü işleme algoritmalarından yararlanılmaktadır. Bu sistemlerin temel fonksiyonları, hareketli nesnenin tespit edilmesi, tanınması ve takip edilmesi olarak listelenebilir. Literatürde, nesnenin iskeletini çıkararak ona ait özelliklerin belirlenmesi tanımlama ve takip uygulamalarının gerçekleştirilmesinde sıklıkla kullanılmaktadır.

Görüntü işlemede iskelet, ilk olarak Blum tarafından 1967 yılında tanımlanmıştır. Blum'a göre, nesnenin sınırlarının tam ortasında bulunan noktalar kümesi şeklin iskeleti olarak tanımlanır ve bu noktalar kümesi nesneyi belirleyen tüm özellikleri içerir. Kısaca iskeletleştirme, nesneye ait verinin gereksiz kısımlarını atarak, nesneye ait özellikleri çok daha az bir veriyle ve doğru bir şekilde tanımlar. Bu nedenle iskeletleştirme kullanan sistemler, daha az kaynak tüketerek daha hızlı çalışabilmektedirler.

İskelet temelli sistemlerin sağlıklı çalışabilmesi için nesnenin iskeletini ifade eden noktalar kümesi iyi tanımlanmalıdır. Doğru bir iskelet tanımı nesnenin geometrik özelliklerini koruyacak şekilde bir piksel inceliğinde olmalıdır. Bugüne kadar Blum'un iskelet tanımına göre bir nesnenin iskeletini oluşturmak için pek çok hesaplama yöntemi geliştirilmiştir. Bu yöntemler; uzaklık döşümü temelli , inceleştirme temelli, morfolojik işlem temelli ve Voronoi diyagramı temelli olarak dört parça altında incelenmektdir.

Bu hesaplama yöntemlerinin bir çoğu mükemmel bir iskelet oluşturamamaktadır. Mükemmel iskeleti oluşturmanın önündeki en büyük engel ise nesnenin sınırlarında oluşan gürültülerdir. Bu gürültüler de iskeleti tanımlayan noktalar kümesine dahil olarak şekli simgeleyen özelliklere zarar verir. Bu durum iskelet bilgisini kullanan sistemlerin kararsız çalışmasına neden olur. Nesneye ait mükemmel iskeletin elde edilmesi ve elde edilen bilginin çeşitli sistemlerde uygulanabilirliği üzerine hala pek çok çalışmalar ve var olan sistemlere eklemeler yapılmaktadır .

Literatürde iskelet bilgisi ile gerçekleştirilen pek çok uygulama bulunmaktadır. Görüntü işlemede en çok karşılaşılan problemlerden biri olan girişim problemini çözmek için birçok sistem tarafından iskeletleştirme kullanılmaktadır. İskelet temelli çeşitli nesne eşleştirme ve takip sistemi de geliştirilmiştir. Nesne eşleştirme sistemlerinde graf eşleştirme olarak bilinen teknikte iskeletten elde edilen nesneye ait özellikler ile bir graf oluşturularak, bu graf bir veri tabanında var olan diğer graflar ile karşılaştırılmaktadır. Nesne takip sistemlerinde ise iskeletin bitiş ve dallanma noktaları takip edilerek tüm nesnenin hareket yönü belirlenebilmektedir. İskeletler parmak izi

ve karakter tanıma sistemlerinde de sıklıkla tercih edilmektedir. Diğer uygulama alanları ise medikal sistemler ve animasyon üretme olarak belirtilebilir. Animasyon sistemlerinde daha çok 3D iskelet bulma yöntemleri kullanılmaktadır.

Yukarıda anlatılan sistemlerin büyük bir kısmı doğru sonuçlar üretmenin yanısıra gerçek zamanlı olarak çalışmalıdır. Bu nedenle bir sistemin performansı, kullanılan algoritmaların hangi ortamda ve nasıl gerçeklendiğine de oldukça bağlıdır. Günümüzde alanda programlanabilir kapı dizileri (FPGA), grafik birim işlemciler (GPU) ve Microsoft tarafından geliştirilen Kinect kamerası ve yazılımı gerçek zamanlı sistemler için sıklıkla tercih edilmektedir. Literatürde FPGA ve GPU üzerinde gerçekleştirilen birçok gerçek zamanlı iskeletleştirme çalışmaları yapılmıştır. Gerçekleştirilen sistemlerin tümü, saniyede işlenen görüntü sayısını artttırmayı ve kullanılan kaynakları azaltmayı amaçlamaktadır.

Mükemmel iskelet tanımı ve gerçek zamanlı çalışma gerekliliğinden yola çıkılarak, bu çalışmada FPGA üzerinde gerçek zamanlı bir iskelet çıkarma işlemi gerçekleştirilerek elde edilen iskelet üzerinden televizyon kontrol sistemlerinde kullanılabilecek bir el takip sistemi geliştirilmesi amaçlanmıştır.

Bu amaca ulaşabilmek için sistemde kullanılacak algoritmanın donanım üzerinde gerçekleştirilmeye uygun ve gerçek zamanlı çalışma hızını yakalayabilmesi gerekmektedir. Literatürde tanımlanan iskelet oluşturma yöntemlerinin bir çoğu mükemmel iskelet tanımını sağlayabilmek için FPGA'de gerçeklenmesi zor kompleks hesaplama teknikleri kullanmaktadır. Bu nedenle çalışmada FPGA üzerinde gerçeklenebilir bir algoritma seçilerek bu algoritmaya sınır gürültülerinin etkisini azaltacak eklemeler yapılmıştır. Bu eklemeler söz konusu algoritmaya ek bir hesaplama yükü getirmemiştir. Sonuçta sınır gürültüsünden bağımsız ve nesnenin geometrik özelliklerini büyük ölçüde koruyan bir iskelet elde edilmiştir. Ayrıca kullanılan yöntem ile bir resim içerisindeki nesnelerin iskeletleri eş zamanlı olarak tespit edilebilmektedir. Bu durum algoritmanın hızı açısından oldukça büyük bir avantaj sağlamaktadır. Elde edilen iskelet, literatürde iskeletin performansını ölçmek için tanımlanan parametreler ile test edilmiştir. Bu şekilde algoritmanın diğer tekniklerle karşılaştırılabilmesine olanak sağlanmıştır.

Elde edilen iskelet sonucu, televizyonlar için uzaktan kontrolü, el hareketleri ile sağlayacak bir el takip sisteminde kullanılmıştır. Bu sistem elin hareketini algılayarak, bu harekete göre televizyon ünitesine çeşitli komutlar göndermektedir. El takip sisteminin gerçekleştirilebilmesi için tanımlanan iskelet üzerinde bulunan bitiş ve dallanma noktaları resim üzerinde 2D konvolüsyon tekniği ile belirlenmiştir. Elde edilen bitiş ve dallanma noktaları takip edilerek tüm elin hareketi belirlenmiş ve komut oluşturulmuştur.

İskelet çıkarma algoritması belirlendikten sonra, bu algoritmanın donanımda hızlı gerçekleştirilmesi için yeni bir mimari önerilmiştir. Sistem üzerinde hareketli nesnenin tespitinden konumunun belirlenmesine kadar olan tüm adımlar gerçekleştirilmiştir. Hareketli nesnenin tespiti için arkaplan çıkarma algoritması tercih edilmiş ve morfolojik operasyonlar ile arkaplan çıkarma algoritmasından elde edilen sonuçlar iyileştirilmiştir. Harekeli nesnenin belirlenmesinden sonra ise sırasıyla iskeletleştirme ve takip algoritmaları gerçeklenmiştir. İskeletleştirme kısmının algoritma adımları

boru hattı mimarisi kullanılarak gerçekleştirilerek sistemin daha hızlı çalışması sağlanmıştır.

Bütün sistem, Digilent Atlys board üzerinde Spartan-6 gibi kaynakları sınırlı olan bir FPGA'de gerçekleştirilmiştir. Giriş resimleri MT9D112 görüntü sensöründen alınmıştır. İskelet çıkarma algoritmasının performansı televizyon kontrolünü uzaktan sağlamayı amaçlayan bir el takip uygulamasında izlenmiştir. Sistemin saniyede işlediği resim sayısı ve kaynak tüketimi belirlenmiştir.

Sonuç olarak takip ve tespit sistemleri için kabul edilebilir sonuçlar üreten bir algoritma gerçek zamanlı çalışacak şekilde donanım üzerine geçirilerek olumlu sonuçlar elde edilmiştir. Hem algoritma hem de donanım tasarımı literatürde bu tarz sistemlerin karşılaştırılmasında en çok kullanılan parametrelere gore değerlendirilerek algoritmik performansı ve gerçekleme performansı açısından karşılaştırılabilir bir çalışma hazırlanmıştır.

1. INTRODUCTION

1.1 Motivation

People benefit from various systems based on image processing technology throughout their daily routine. Some noteworthy systems would include biomedical applications, traffic monitoring systems, animation generation, human-computer interaction systems. The general functions of these systems are tracking, recognizing, matching, etc. Inspired from this fact, this thesis pursues a real time implementation of skeletonization which forms the backbone of many tracking and matching applications.

Skeleton or medial axis transform was first introduced by Blum in 1967 [3]. The skeleton of a shape is defined as a set of points that lie midway between object boundaries. The skeletonization process eliminates the redundant part of the object and it gives the opportunity to analyse the shape with significantly smaller amount of data. Consequently, it saves computation time and reduces resource utilization which are regarded as the main objectives for the design of such systems.

Different skeletonization techniques have been developed to produce a skeleton according to Blum's definition. These techniques can be classified into thinning, distance transform, voronoi and morphological methods. Each skeleton computation method generate a skeleton with different features. Selection of the computation method is crucial since only the proper calculation technique improves the performance of the application.

There are a variety of applications reported in literature that stems from skeletonization. Many algorithms have been developed for shape recognition systems based on graph or tree representation of features extracted from the skeleton [5], [6], [7], [8], [9], [10], [11]. Skeletonization can also be used to handle occlusions. Chen introduced a method to detect and segment occluded vehicles for traffic monitoring systems using skeleton features [12]. People-vehicle classification system in [13] is another example of skeleton based recognition systems.

Object tracking applications take advantage of skeletonization as well. Jimenez in [14] represents a hand tracking mechanism using skeletal features where the motion of a hand is tracked, successfully handling the occlussion problem. Animation generation can also be considered among the applications that benefit from skeletonization [2]. Furthermore, variety of systems are proposed to verify fingerprints with skeletonization [15], [16], [17], [18]. Likewise, skeletonization is exploited excessively in medical applications such as in [19] and [20].

The above mentioned applications often require real time processing capabilities. Therefore, the preference of an algorithm for a system highly depends on the real time performance of the algorithm. Recently, implementation of image and video processing algorithms on Field-Programmable Gate Arrays (FPGA) have received considerable attention due to their capability to meet high performance and low power requirements. Accordingly, FPGAs are frequently selected as the instrument for the implementation of skeletonization algorithms. In addition to FPGAs, Graphics Processing Units and Microsoft Kinect Camera and Software are also preferred for such systems.

The computational complexity of the skeletonization algorithms highly increases to reach a performance close to perfect skeleton. Perfect skeleton should preserve geometrical properties of the shape and it should be one pixel thick. This complexity makes it impossible for the systems to cope with real time requirements. It is obvious that there is a trade of between generating a perfect skeleton and satisfying real time capabilities for the systems. So, skeletonization with good performance without increasing the complexity for the real time systems is a challenging task. Due to this fact, this thesis investigates a method to improve both real time and algorithmic performance of the existing skeletonization architectures on FPGA for matching and tracking systems. Our method aims to produce skeletons, which are one pixel thick, invariant to the orientation, insensitive to noise and independent of the shape, while reducing the resource utuilization and increasing the number of processed frames per second.

Iterative thinning and distance transform based methods are the most widely used skeletonization techniques. To date, many thinning and distance transform based implementation methodologies have been developed for image skeletonization on FPGAs. Bourbakis et al. [21] proposed an application specific array processor for high speed thinning. Lopich and Dudek presented a thinning architecture that uses asynchronous cellular processor array [22]. Thinning based image skeletonization on FPGA is also submitted in [23], [24], [25]. Thinning techniques are highly suitable for VLSI implementation; however they are dependent on the shape which makes them unreliable for shape matching systems [26].

Distance transform based algorithms are distinguished among themselves according to distance transform metric. Ranganathan represented a method using cityblock distance transform [27]. Skeletons extracting from a distance map with cityblock metric is not invariant to the orientation. Thus, the method fails under rotations and translations of the image. Euclidean distance transform is the ideal candidate of all distance transforms for skeletonization based shape matching systems since it is independent of shape and invariant to orientation. On the contrary, skeletonization based on Euclidean distance transform is not a convenient solution for VLSI implementation because it requires large look up tables for computation of Central Maximal Disks (CMDs) [28]. However, Sudha suggested a skeletonization technique for FPGAs based on Euclidean distance transform [29]. This work calculates the euclidean skeleton and the distance values simultaneously and the proposed system can process approximately 30 frames per second. In this thesis, an integer approximation to Euclidean distance named Chamfer(3,4) is preferred to reduce the resource utilization and increase frame rate of the system since it is more feasible for FPGA realization. It also preserves shape independency and orientation invariance [30].

Extraction of skeletal points is the stage which follows distance transform operation. This stage requires complex mathematical computations making it hard for FPGA implementation. Besides, this stage urges a subsequent stage which is the removal of unwanted branches of the skeleton, called pruning. Pruning brings in extra complexity to the hardware implementation and it consumes extra process time. Chang et al.

[31] suggests a design to overcome this mathematical complexity but the unwanted branches are still observed in the output skeleton.

1.2 Contribution

The main contribution of this work, providing the functionality of pruning with an extension to Chang's work without introducing additional computation time on Chamfer(3,4) distance map. In this way, the proposed system is insensitive to the boundary noise, independent of shape and invariant to the orientation with higher performance and lower resource utilization than real time requirements.

Also, a fully pipelined architecture is proposed to implement the extended skeleton extraction algorithm on hardware. Complete system is developed on Digilent Atlys board with Spartan-6 FPGA to test the proposed skeletonization architecture. The frames are captured from CMOS image sensor with an integrated advanced camera system MT9D112. The performance is observed on a hand tracking application. Hand gesture are widely used in human-computer interaction systems [32] and can be comprehended easily using the skeletal features [14]. In this thesis, motion of the hand is tracked for remote control for television sets exploiting branch points and end points of the skeleton.

1.3 Organisation of Thesis

This thesis is organized in five chapters including the Introduction chapter. Chapter 2, introduces the concept of skeletonization, covers all the existing skeleton extraction methods in literature and compares the methods with each other. The theory of the proposed skeletonization system is described in Chapter 3. This chapter also presents the performance evaluation criteria for the system and demonstrates MATLAB simulation results. Chapter 4 is dedicated to the FPGA implementation of the system and experimantal results. Finally, Chapter 5 concludes this thesis.

2. SKELETONS IN DIGITAL IMAGE PROCESSING

Skeleton is a crucial shape representation technique in image processing. It is the output of the skeletonization process. Skeletonization process can be defined as a pixel transformation. This transformation converts most of the original foreground pixels to background pixels while preserving the skeletal residue on the binary image and the shape is represented as a 1D linear data by eliminating redundant part. The skeleton of letter T can be seen in Figure 2.1 as an example. Preserved skeletal residue is determined based on skeleton definitions which are described in the following subsection. Also, this chapter explains the representation and extraction methods of the skeleton in detail.

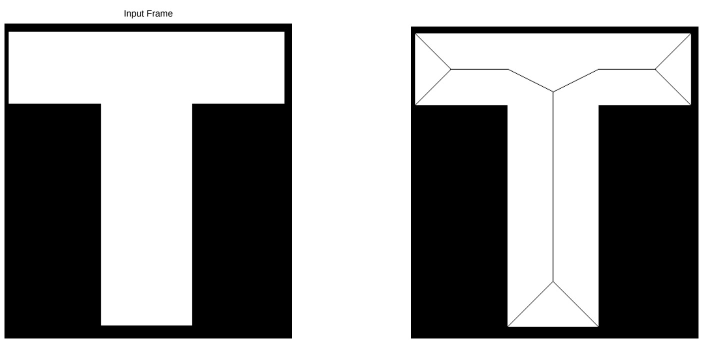


Figure 2.1: Binary Image of Letter T and its Skeleton.

Notation :

In this thesis, a shape or object on the binary image is denoted with *O*. This work focuses on two dimensional images so $O \subset \Re^2$. The boundary of the object is shown with $B \equiv \partial O$. *MAT*, short for medial axis transform, represents the skeleton of the shape.

2.1 Definitions and Properties of Medial Skeleton

The medial skeleton or medial axis was first described by Blum in 1967 [3]. According to Blum's basic medial axis definition, the skeleton of a 2D shape is the loci of points equidistant from the contour. In this section, two basic analogies to find Blum's medial axis, stated in [28], are examined in detail.

2.1.1 Maximally-inscribed balls

This methodology places a pixel-sized ball on each boundary pixel. The process starts with expanding the ball towards inside of the boundary, the starting boundary pixel being tangent to the ball. The ball continually expands until another boundary pixel becomes tangent to the ball. The locus of the center of the ball with at least two boundary pixels tangent to the ball is classified as a medial axis point.

Definition : The medial axis transform of the object is the set of centers of maximally-inscribed balls and radii of the all maximally-inscribed balls in the object [33].

Mathematically, $[M,R] = MAT(O)$ where O is the object, M is the center points of maximally inscribed balls and R is the radius of the corresponding spheres.

The skeleton of a rectangular object which is defined via maximal discs can be observed in Figure 2.2. As shown in the figure, A and B are skeleton points, but point C is a foreground pixel.

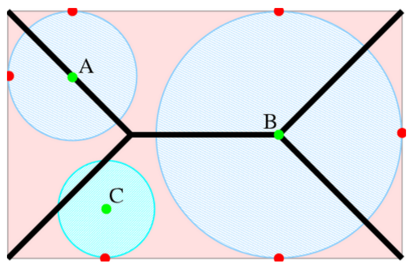


Figure 2.2: Skeleton of a Rectangle via Maximally-Inscribed Balls [2].

2.1.2 Grassfire analogy

In this approach, the boundary of the object is accepted as the starting point of a fire. The fire spreads along the normals **n** with uniform speed. When the fire expands enough, it meets with the other fire which is started at another part of the boundary and these wavefronts quench each other. These quench points are equidistant from two different parts of the boundary as defined in Blum's definition. Therefore, the loci of quenchpoints constitute the medial axis. The generated wavefronts are shown in Figure 2.3.

In grassfire analogy, the quenchpoints are always equidistant from the boundary. As a result, medial points have at least two closest points on the object boundary (B).

Definition : Medial axis is the set of locations M internal to the object with more than one corresponding closest point and their distance R from the boundary B [34].

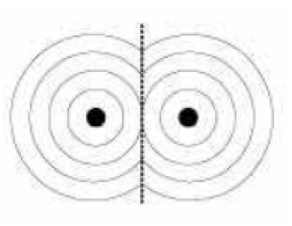


Figure 2.3: Wavefronts generated by a two point Grassfire Excitation [3].

These definitions are the most well-known definitions for searching Blum's medial axis. Some definitions other than Blum's are also stated in literature. Brady's Smooth Locus of Symmetries (SLS) and Leyton's Process Induced Symmetric Axis (PISA) are worth mentioning for a medial skeleton definition [28].

2.2 Skeleton Extraction Methods

The observed skeleton, which is generated via a skeletonization process, should provide some desirable properties. These properties were pointed out in [26] and [35]:

1- The skeleton should preserve the topological information of the original object.
2- The skeleton must be centered within the object boundary.
3- The skeleton should be stable under small deformations.
4- The skeleton should contain the centers of maximal discs, which can be used for reconstruction of the original object.
5- The skeleton should be invariant under Euclidean transformations, such as rotations and translations.
6- The skeleton should represent significant visual parts of objects.

7- The skeleton should be as thin as possible.
8- The output skeleton must have the same connectivity as the original shape.

In discrete space, all of the above properties are mutually exclusive. So it is not possible to accommodate all the properties together. Every skeletonization algorithm can provide a different subset of the desirable properties and the proper skeletonization method should be selected depending on the application. The skeletonization methods can be classified into four types:

- Distance Transformation,
- Voronoi Diagram,
- Thinning Techniques,
- Mathematical Morphology.

2.2.1 Distance transformation

A distance transform converts a binary image consisting of feature and non-feature pixels into an image where each pixel value denotes the distance to the nearest feature pixel [36]. In other words, the distance transformation calculates the distance to the nearest background pixel for each pixel of the shape on the binary image.

There are five distance transformation metrics which are cityblock, chessboard, Chamfer 3-4, Chamfer 5-7-11 and Euclidean [36].

Cityblock technique measures the path between pixels based on a 4-connected neighborhood. In 2D, the cityblock distance between pixels located at (x_1,y_1) and (x_2,y_2) is calculated as:

$$distance_{cityblock} = |x_1 - x_2| + |y_1 - y_2|. \tag{2.1}$$

Chessboard transformation measures the path between pixels based on a 8-connected neighborhood. In 2D, the chessboard distance between pixels located at (x_1,y_1) and (x_2,y_2) is evaluated as:

$$distance_{chessboard} = max(|x_1 - x_2|, |y_1 - y_2|). \quad \textbf{(2.2)}$$

The *Euclidean* distance is the shortest distance between two pixels. In 2D, the Euclidean distance between pixels located at (x_1,y_1) and (x_2,y_2) is defined as:

$$distance_{euclidean} = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}. \quad \textbf{(2.3)}$$

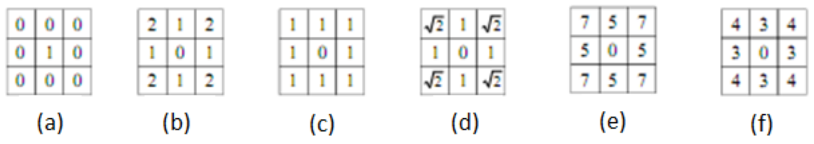


Figure 2.4: Masks of Different Metrics (a) Input (b) Cityblock Distance (c) Chessboard Distance (d) Euclidean Distance (e) Chamfer 5-7-11 (f) Chamfer 3-4.

Integer number values are frequently preferred to real number values in distance transforms [37]. *Chamfer 3-4* and *Chamfer 5-7-11* metrics, known as the classical Chamfer metrics, approximate the Euclidean distance to integer distance values. *Chamfer 3-4* and *Chamfer 5-7-11* metrics look at 3x3 and 5x5 neighborhood respectively. The masks for different metrics can be observed in Figure 2.4. The values of the chamfer masks are detremined to approximate the $\sqrt{2}/1$ ratio in Euclidean mask [30]. For example $\sqrt{2}/1$ = 1.41 is approximated by 4/3 = 1.33 for Chamfer 3-4 and it is approximated by 7/5 = 1.4 for Chamfer 5-7-11. There are other masks to approximate Euclidean mask; but according to Hajdu et al. [30] classical Chamfer distances are considered as the best approximates. Figure 2.5 shows the distance transform of an input image of 7x7 square for all types of transformation metrics.

The skeleton of the shape can be determined using one of these distance transformation techniques. Skeletonization using distance map is based on the following idea: When a progression is started from each boundary point of the shape, the distance map has increasing values until it reaches a ridge. These ridge points constitute the medial axis of the shape. Figure 2.6 shows a cityblock distance transformation result. Points with distance values written in italic are the ridge points of the transform.

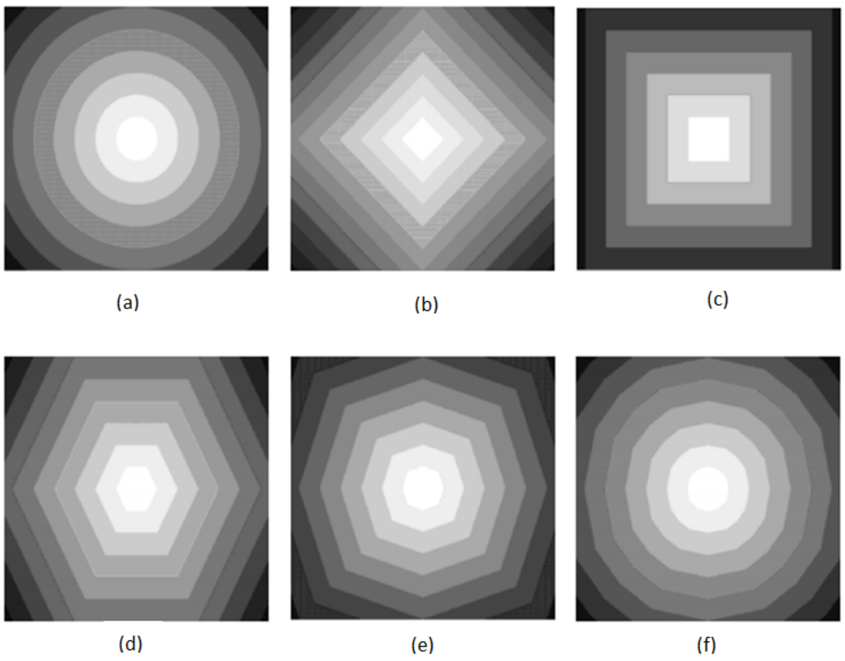


Figure 2.5: Severel Types of Distance (a) Euclidean (b) CityBlock (c) Chessboard (d) Hexagonal (e) Chamfer 3-4 (f) Chamfer 5-7-11.

4	3	2	1	2	3	4
3	2	1	*0*	1	2	3
2	1	*0*	1	*0*	1	2
2	1	*0*	1	1	*0*	1
1	*0*	1	2	2	1	*0*
1	*0*	1	2	3	2	1
0	1	2	3	4	3	2

Figure 2.6: Ridge Points of the Distance Transform.

The algorithms based on the ridges of the distance map, can ensure the accurate localization of skeletal points but neither connectivity nor completeness [38]. These methods add a linking process as final step to handle the connectivity problem. Di Baja proposed a method in 1994 [1]. In his work, first the distance transform is calculated. Local maxima and saddle points of the distance map are defined as ridge points as the second step. Third step is to grow connected paths in the direction of maximal gradient. Finally the holes are filled. Another example of this approach is expressed in [39].

2.2.2 Voronoi diagram

In mathematics, a Voronoi diagram is a way of dividing space into a number of regions [40]. The regions are called cells. Sample points (generating points) are generated from the boundary B of the object/shape O to split the space into cells. Each cell contains exactly one sampling point and the locus of all points which are nearer to the corresponding sampling point. The partition of the space is the Voronoi diagram. The Voronoi diagram converges to the skeleton when the density of the sample points goes to infinity [2].

Figure 2.7 demonstrates computation of the skeleton of a rectangle using Voronoi diagram. First, some boundary points are specified as generating points, then the skeleton is approximated by a subgraph of the Voronoi diagram. The skeleton can be identified with the *red line* in Figure 2.7.

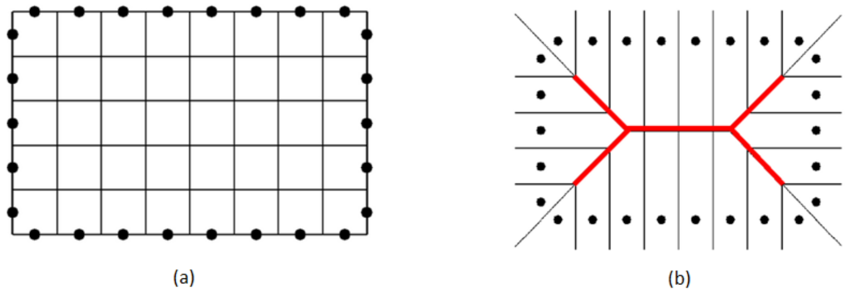


Figure 2.7: Voronoi Skeleton of a Rectangle (a) Generating (sample) Points (b) Approximating Skeleton [2].

Several Voronoi diagram based skeletonization techniques were developed. All these algorithms preserve topology of the shape very well and supply most of the desirable properties of the skeleton. On the contrary, it is a computationally expensive process, especially for large objects [2].

2.2.3 Thinning techniques

Thinning is an algorithm that removes pixels from boundary of the object iteratively until the skeleton of the object remains. Thinning methodologies usually preserve topology of the object. Also, they provide exactly one pixel thick skeleton in the middle of the image. A demonstration of a thinning process can be seen in Figure 2.8.

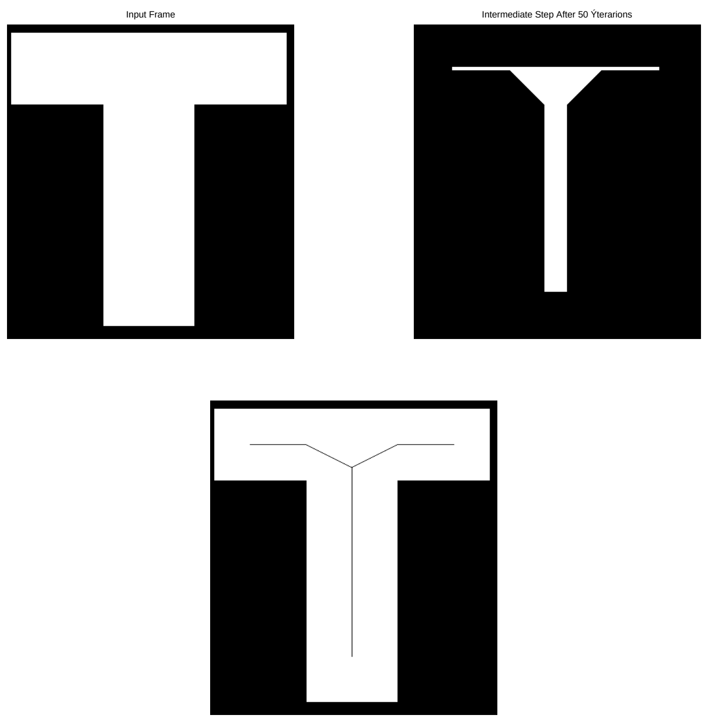


Figure 2.8: Thinning Process.

In thinning techniques, iterative deletion of contour pixels can be done either sequentially or in parallel way [41]. Sequential thinning algorithms delete single pixel at a time and they preserve the topology of the shape. Parallel thinning algorithms examine all pixels to delete in a single iteration based on the previous iteration result. Therefore, parallel thinning methods remove many points at a time. This type of deletion can damage the topology of the object. For all thinning methods, it's important to determine a good stop criteria for the iteration process [35].

Thinning methods are sensitive to boundary noise and they usually fail to localize the accurate skeletal position. Another problem of thinning methodologies is that the rules for deleting a pixel from the boundary highly depends on the type of the object. Also, it is a time consuming process [26].

2.2.4 Mathematical morphology

Mathematical morphology is a non-linear theory for image processing based on set theory. It yields a non-linear method for geometry based processing [42]. A structuring element is defined for a morphological process. The value of each pixel in the output image is based on a comparison of the corresponding pixel in the input image with its neighbors in the structuring element (SE). Structuring element is defined according to origin of SE and position of elements belonging to SE. The shape and size of SE must be adapted to the geometric properties of the objects [43]. Figure 2.9 gives different examples for structuring element.

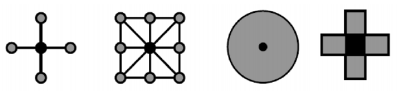


Figure 2.9: Samples of Structuring Elements.

There are four basic morrphological operators.

$\ominus Erosion$

$\oplus Dilation$

$\circ Opening$

$\bullet Closing$

Dilation add pixels to the boundaries of objects in an image, while erosion removes pixels on object boundaries.

Opening is an erosion process followed by dilation using the same structuring element. It eliminates protrusions and breaks connections. It is defined as:

$$A \circ B = (A \ominus B) \oplus B. \tag{2.4}$$

where A is the input image and B is the SE.

Closing is a dilation process followed by erosion. It eliminates small holes and fills gaps. It is defined as:

$$A \bullet B = (A \oplus B) \ominus B. \tag{2.5}$$

where A is the input image and B is the SE.
Both opening and closing processes smooth boundary of the shape. Figure 2.10 displays an opening operation on a binary image using 3x3 structuring element.

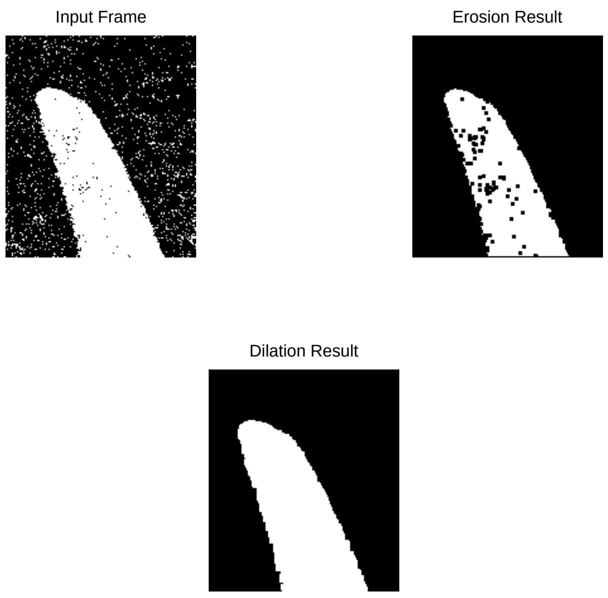


Figure 2.10: Opening Process.

According to [42], the skeleton of a shape can be calculated by means of binary morphological operators. In this theory, the skeleton of an object is expressed as:

$$S(O) = O(\ominus)rB - [(0 \ominus rB) \oplus drB]. \qquad \textbf{(2.6)}$$

where rB is the topologically open disc and drB is the topologically close disc.

The set $O \ominus rB$ represents the portion of grassfield not yet burned by the fire at time $t = r$ in the grassfire model. The set $0 \ominus rB \oplus dr$ represents the points at which the fire does not quench at time $t = r$. Therefore, the difference between the above sets gives the skeleton points [42].

Usually morphological operation based skeleton search can localize the accurate skeleton, but may not guarantee the connectivity of the skeleton [35].

2.2.5 Comparison of skeleton extraction methods

As mentioned above, the skeleton calculation methods supply a set of desirable properties. Table 2.1 compares the skeleton extraction methods in terms of connectivity, centeredness, thinness and transformation invariance criteria. Distance tranform based methods can produce the skeleton in the middle of the object and they

Table 2.1: Comparison of Skeleton Extraction Methods.

Method	Connectivity	Centeredness	Thinness	Trans. Inv.
Distance Transform	No	Yes	No	Yes
Voronoi	Yes	Yes	?	?
Thinning	Yes	No	Yes	No

are invariant under transformations; but these methods have connectivity and thinness problems. On the contrary, thinning methodologies preserve topology perfectly and produce one pixel thick skeleton; however the skeleton may not be in accurate location and these methods do not accommodate the invariance criteria under transformations. The skeletons obtained from the Voronoi diagram, preserve topology perfectly as the thinning techniques and are located in the middle of the shape as the distance transform based methods.

2.3 Classification Of Skeleton Points

The points of a skeleton are classified as either normal points or branch points or end points.

If a skeleton point has two active neighbors, it is called a normal point. If a skeleton point has three or more active neighbors, it is called a branch point. If a skeleton point has a single active neighbor, it is called a end point.

The branch points (red points), end points (blue points) and normal points (white points) of a skeleton are shown in Figure 2.11.

2.4 Skeleton Pruning

All the skeleton extraction methods which are mentioned above produce skeletons with unwanted branches. These unwanted branches are generated due to the boundary noise of the shape as can be observed in Figure 2.12.

To overcome the unfavorable effect of boundary noise, some skeletonization methods are followed by a pruning operation as a post-process and they produce acceptable results [35]. Alternatively, the boundary of the shape can be smoothed prior to skeletonization [44]. Altough, this solution is weak in generating skeletons in the

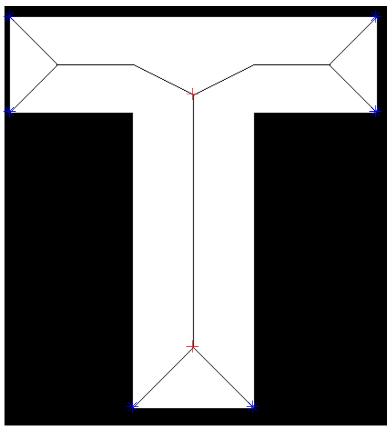


Figure 2.11: Branch, Normal and End Points of Letter *T*.

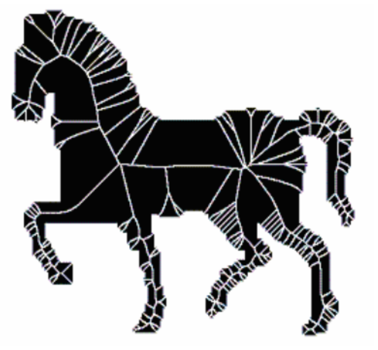


Figure 2.12: Skeleton with Unwanted Branches.

accurate location [44]. As the third approach, pruning functionality can be assured by a modifying the standard skeleton computation methods [39]. This thesis focuses on the third approach.

3. REAL TIME SKELETONIZATION ALGORITHM AND SIMULATIONS

Skeleton computation methods, which are discussed in the previous chapter, can not satisfy all the desirable properties that a skeleton should have. Therefore, it is a crucial step to choose the most appropriate computation technique to profit from the advantages of skeletonization and improve the performance of the application. In this thesis, the main goal is to design a real time skeletonization system on FPGA for matching and tracking applications. Two points must be considered in choosing the computation method:

- The calculation procedure must be suitable for hardware implementation.
- The method must produce satisfying results for matching and tracking applications, i.e., the produced skeleton must be connected, it must preserve the end points of the shape, it must be invariant under translations and rotations.

Skeleton extraction methods are re-examined under these conditions to choose the most appropriate method for our system. Thinning techniques guarantee the connectivity and they are suitable for VLSI implementation; but they fail under translations and rotations of the shape which makes them unreliable for matching systems. On the other hand, skeletonization using Voronoi diagrams meet all the desirable properties for matching and tracking systems. Unfortunately, it is a computationally expensive process, which is not convenient for real time implementation.

Distance transform based algorithms are distinguished among themselves according to their distance transform metric. Distance transform based methods that use Euclidean or Chamfer metrics can generate invariant skeletons under rotation and translation; tough Chamfer metric based ones are favorable for hardware implementation. However, connectivity and completeness of the skeleton can be a problem for distance transform based algorithms. As a result, a Chamfer 3-4 distance transform based technique is developed paying attention to the connectivity and completeness criteria for a real time image skeletonization system. Skeletonization using distance transformation can be computationally complex and the algorithm can include high order derivatives which are hard to implement on FPGA. Chang et al. [31] proposed a novel method to find ridge points on distance transform. This method is simple; but an efficient solution for defining skeletal residue in accurate location. It only checks the neighbors for each pixel and decides if it is a skeletal point or not. Nevertheless, unwanted branches are generated in addition to main skeleton due to boundary noise. This means that, a seperate pruning step is needed after exracting the skeleton which brings in extra complexity to the algorithm as well as extra process time consumption. In this work, the functionality of pruning is provided with an extension to Chang's work without introducing additional computation time. Eliminating the necessity of a seperate pruning step also makes the algorithm more suitable for FPGA implementation. The proposed framework is named "extended ridge point detection algorithm" hereafter. The performance of the produced skeleton using the extended ridge point detection algorithm is observed on a hand tracking application. In this thesis, motion of the hand is tracked for remote control for television sets exploiting branch points and end points of the skeleton. The end points and branch points are designated using a 2D convolution.

The following section describes Chang's method to extract skeletons from distance maps and explains the proposed extension to Chang's work. Performance evaluation parameters and the performance results of the extended ridge point detection algorithm regarding these parameters are presented in Section 3.2. Matlab simulation results of the algorithm for several input images are also included in this section. Section 3.3 introduces the 2D convolution based end point and branch point designation for the hand tracking application and contains Matlab results of this technique.

3.1 Skeleton Extraction Algorithm

The following subsection covers the steps of the Chang's ridge point detection algorithm and its drawbacks. Subsection 3.1.2 defines the suggested modification, which is named as "extended ridge point detection algorithm", to Chang's work.

3.1.1 Ridge point detection algorithm

Ridge point detection algorithm that is used in this work is a gradient-based method. For a point to be a ridge point on a distance map, it must be local maxima in some direction. If the point is local maxima in a direction, the two opposite neighbors of the point in that direction has smaller distance values than the corresponding pixel. So, the ridge point generates a sign barrier between the two opposite neighbors on the direction line. All the ridge points on distance map are determined by examining the map in four orientations of 0, 45, 90, 135 degrees as shown in Figure 3.1. In the figure, each ridge point intersects at least one scanline and generates a sign barrier. However, examining the distance map in four orientations is hard to realize. Chang et al. [31] reported that, it is sufficient to examine two orthogonal scanlines on distance map to find all the ridge points. In other words, scanning the distance map from left to right and top to bottom specifies all the ridge points.

When a search is started on a scanline, several sign change patterns, which are the indications of a ridge existence, are encountered. For the left to right and top to bottom scan, there are only six possible patterns: +-, -+, +0, 0+, 0- and -0, where + represents a vector positive direction on a scanline, - represents a vector negative direction on a scanline and 0 means a zero vector.

Pattern +- is the most prominent indication of a ridge point existence. If the ridge intersects the scanline at an integer coordinate point, the +0- pattern is generated instead of +- pattern. Patterns +0 and +0- are called *strong* ridge existence indicators.

Pattern -+ is an indication of a valley. It occurs in two ways. First way, if two ridges exist in such a way so that they enclose two neighboring points. This case is ignored, since it is not a ridge point indicator. Second way which -+ pattern occurs is when two tapering shapes meet at their closing ends. In this case the ridge appears as a +- pattern on the other orthogonal scanline. Thus, it can also be ignored.

Pattern +0 occurs in two ways as well:

- at the aliased edge of a shape,
- at the beginning of a plateau on distance map.

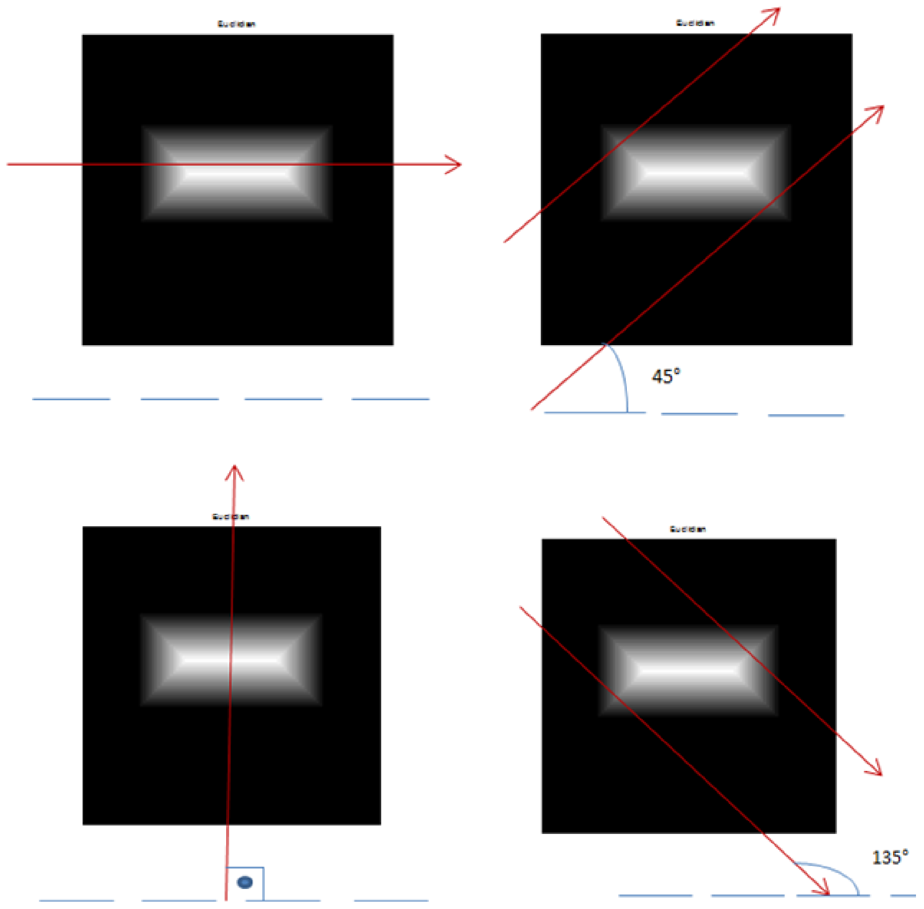


Figure 3.1: Scanning the Rectangle for Four Orientations (0, 45, 90, 135 Degrees).

If this pattern occurs in first way, it is not accepted as an indicator since it exhibits spurious skeleton branches on the aliased edge. If the pattern occurs in second way as a plateau, it indicates a ridge existence. When it occurs on a plateau, it must always occur in pairs with one of the +-, 0-, +0, +0- patterns on the other scanline. If it is not coupled with other patterns, this pattern is called a *weak* indicator. Otherwise, it is called a *good* indicator of ridge point existence. Pattern 0- is treated similar to +0 pattern.

Pattern 0+ indicates the leftmost or topmost edge of the object, if it is the first sign barrier detected on the relevant scanline. So, the 0+ pattern should not be the first sign barrier on the scanline to be an indicator of ridge point existence. This pattern can be extended either as -[0...]+ or +[0...]+. The extended pattern -[0...]+ indicates the existence of a valley or a basin and it is not accepted as a ridge point for the same reasons that are described for pattern -+. For the other extended pattern +[0...]+, the '+' at the end indicates the start of a new object, the '+' at the beginning shows that the point is already inside another shape. Therefore, pattern 0+ is merely an indication of the start of a new subshape. The extended patterns are not accepted as ridge points, because they can be detected by other patterns.

The pattern -0 can be explained in a similar manner as pattern 0+ . It indicates the rightmost or bottommost edge of the object, if it is the last sign barrier detected on the relevant scanline. So, the -0 pattern should not be the last sign barrier on the scanline to be an indicator of ridge point existence. This pattern can be extended either as -[0...]+ or -[0...]-. The extended pattern -[0...]+ indicates the existence of valley or basin and it is not accepted as a ridge point for the same reasons that are described for pattern -+. For the other extended pattern -[0...]-, the '-' at the beginning indicates the end of a new object, the '-' at the end shows that the point is already inside another shape. Therefore, pattern -0 is merely an indication of the end of a subshape. The extended patterns are not accepted as ridge points, because they can be detected by other patterns.

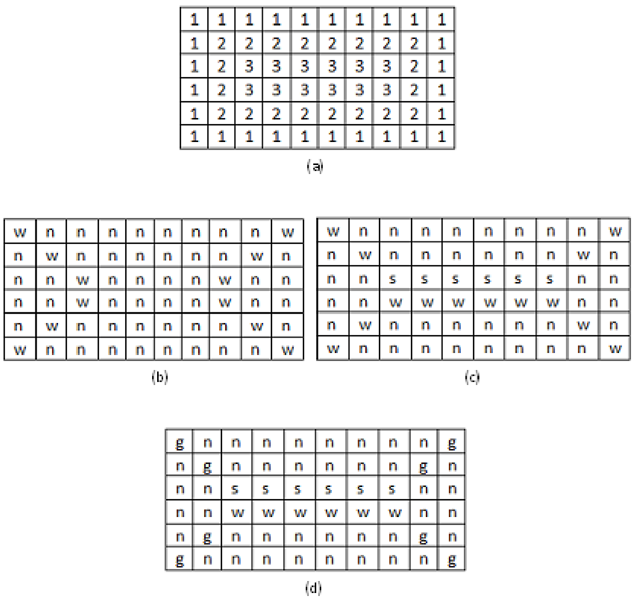
After examining all the possible patterns and their extensions, the four patterns are accepted as indicators of ridge point existence : +-, +0-, +0, 0-. These patterns are called *prominent sign barriers*. The patterns +0 and 0- must be paired with one of these patterns on the other scanline. With all required patterns specified, ridge point detection algorithm can be described in three steps:

1. Distance Transform is calculated.
2. Relative location of a point is computed for each scanline using Equation (3.1) and Equation (3.2).
3. The *prominent sign barriers* are searched on the image by scanning from left to right using N_y and scanning from top to bottom using N_x. When the patterns are detected on the scanline, they are labeled as either *weak* or *good* or *strong* or *none*.

$$N_x \equiv sign(D(x+1,y) - D(x,y)) \qquad \textbf{(3.1)}$$

$$N_y \equiv sign(D(x,y+1) - D(x,y)) \tag{3.2}$$

Figure 3.2 shows the output of the ridge point detection algorithm for a 6 pixel x 10 pixel sized rectangular shape for each scan. *Weak* points are labeled with 'w', *good* points are labeled with 'g', *strong* points are labeled with 's', *none* points are labeled with 'n'.



1	1	1	1	1	1	1	1	1	1
1	2	2	2	2	2	2	2	2	1
1	2	3	3	3	3	3	3	2	1
1	2	3	3	3	3	3	3	2	1
1	2	2	2	2	2	2	2	2	1
1	1	1	1	1	1	1	1	1	1

(a)

w	n	n	n	n	n	n	n	n	w
n	w	n	n	n	n	n	n	w	n
n	n	w	n	n	n	n	w	n	n
n	n	w	n	n	n	n	w	n	n
n	w	n	n	n	n	n	n	w	n
w	n	n	n	n	n	n	n	n	w

(b)

w	n	n	n	n	n	n	n	n	w
n	w	n	n	n	n	n	n	w	n
n	n	s	s	s	s	s	s	n	n
n	n	w	w	w	w	w	w	n	n
n	w	n	n	n	n	n	n	w	n
w	n	n	n	n	n	n	n	n	w

(c)

g	n	n	n	n	n	n	n	n	g
n	g	n	n	n	n	n	n	g	n
n	n	s	s	s	s	s	s	n	n
n	n	w	w	w	w	w	w	n	n
n	g	n	n	n	n	n	n	g	n
g	n	n	n	n	n	n	n	n	g

(d)

Figure 3.2: Ridge Points on Distance Transform Image (a) Distance Transform Image (b) Result of y axis Scan (c) Result of x axis Scan (d) Output Image of the Ridge Point Detection Algorithm

First of all, only strong and good points are accepted as skeleton points. However gaps may occur on the skeleton due to discrete nature of the map. For a connected skeleton, all the skeleton pixels must have at least two neighbors. Therefore, a linking process is required to fill the gaps. In this algorithm, the points constituting the gaps are usually labeled as *weak* points. These points help us to make the correct link between unconnected skeletal parts. So, all the skeleton points, which have less than two neighbors, are linked to the nearest *weak* point among 8 neighbors. If the relevant point does not have a *weak* ridge neighbor, then it is connected to the point with the maximum gradient. The linking process continue until all end points are connected to a branch point or a border point of the shape.

The performance of the algorithm described above highly depends on the distance metric, which is used in distance transformation step. The algorithm must satisfy the invariance criteria under rotation and translation for our system. However, all distance metrics can not produce the same results under translations and rotations. This feature can only be guaranteed with a distance map that is formed using the Euclidean or Chamfer metrics. Distance transformation with Euclidean metric is hard to implement on hardware. Chamfer metrics are good solutions for VLSI implementations and they can produce results very close to Euclidean metric as described in Chapter 2. Chamfer 3-4 metric is used for the skeletonization system which is introduced in [1] and it produced satisfying results. Therefore, the ridge point detection algorithm is run on a distance map that is extracted using Chamfer 3-4 metric.

The ridge point detection algorithm is implemented in Matlab environment and tested on various images. Figure 3.3 displays the output images. When the output results are examined, it is observed that *strong* labeled ridge points can produce unwanted skeleton branches in addition to the main skeleton. So, a separate pruning algorithm is required following the ridge point detection algorithm.

Instead of applying an independent pruning algorithm as another step, the ridge point detection algorithm is extended by adding new rules for defining a *strong* labeled ridge point as a skeleton point. The extended method is expressed in the next subsection in detail.

3.1.2 Extended ridge point detection algorithm

This subsection introduces the extended ridge point detection to produce pruned, connected and one-pixel thick skeleton.

As stated in the previous subsection, *strong* labeled ridge points can produce unwanted skeleton branches in addition to the main skeleton. Therefore, detecting +- or +0- pattern is not enough to accept a point as a *strong* ridge. When +- or +0- pattern is encountered on a scanline, it is accepted as *strong* ridge existence indicator if and only if the ridge point is local maxima within 3x3 neighborhood and within object boundaries on that scanline. Also, the distance value of the point must be higher than a predetermined threshold *T*. This threshold value prevents the points, which are too

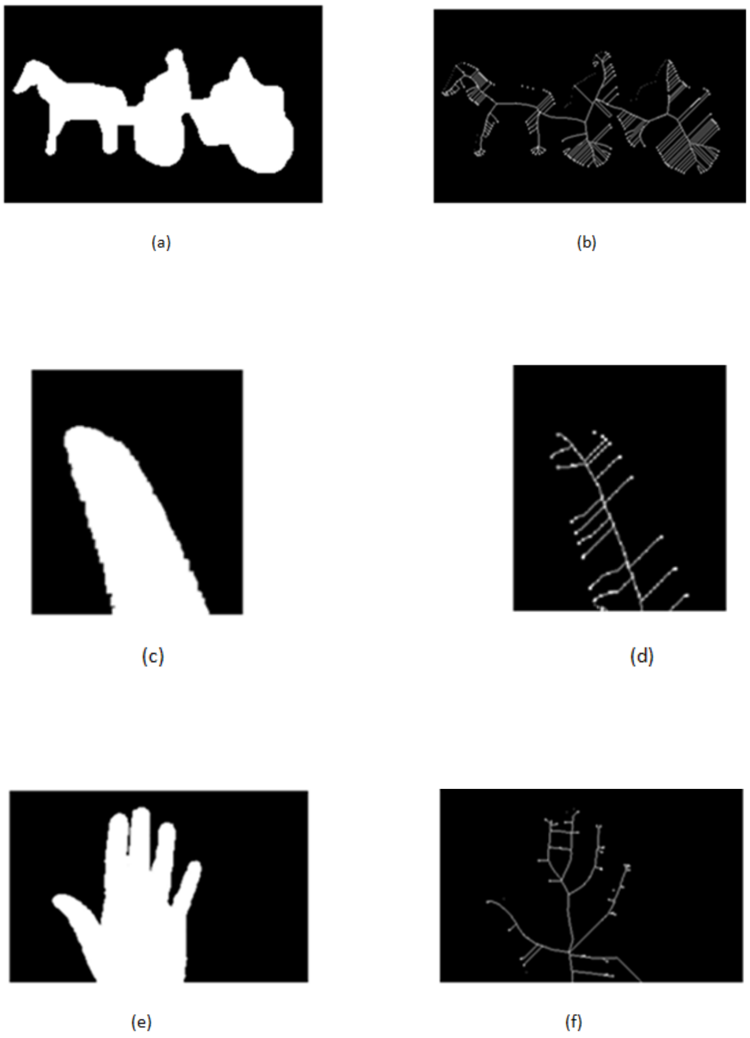


Figure 3.3: Test Results : (a), (c), (e) Input Binary Images (b), (d), (f) Skeleton Outputs with Ridge Point Detection Algorithm.

close to the border of the shape and satisfy the above conditions, to be accepted as a ridge point.

The conditions for local maxima on weighted distance transforms are represented as:

$$D_{x,y} + a \geq D_{x+k,y+l}, \tag{3.3}$$

$$D_{x,y} \geq D_{x+q,y+t}, \tag{3.4}$$

$$D_{x,y} \geq T, \qquad \textbf{(3.5)}$$

where k,l $\in \{-1,1\}$, q,t $\in$ {pixels within boundaries}, D indicates the distance transform value and a=3 and T=7 for Chamfer 3-4 metric.

In weighted distance transforms, a pixel can be a local maxima even if it has neighbors with higher values [1]. So, local maxima for weighted distance transforms is specified correctly by adding *a* to the distance transform value when comparing it with the values of its neighbors.

To sum up, the rules for classifying the points as *strong*, *good*, *weak* or *none* are changed as:

If a point creates +- or +0- pattern and if the conditions in Equations (3.3), (3.4), (3.5) are satisfied, the point is marked as a *strong* ridge. If the conditions are not satisfied, then the point is labeled as a *weak* ridge.
If +0 or 0- pattern is detected, paired with one of the patterns on the other scanline, and the point satisfies the condition in Equation (3.5), the point is marked as a *good* ridge. If +0 or 0- pattern is not paired with any patterns on the other scanline, the point is marked as a *weak* ridge.
If a point does not create one of +-, +0-, +0 and 0- patterns; but it is a local maxima, then it is labeled as a *weak* ridge, too.

Accordingly, the steps of the algorithm can be redefined:

1. Distance Transform is calculated using Chamfer 3-4.
2. Relative location of a point is computed for each scanline using Equation (3.1) and (3.2).
3. Local maxima is computed for each scanline using Equation (3.3) and (3.4).
4. The *prominent sign barriers* are searched on the image by scanning from left to right using N_y and scanning from top to bottom using N_x and the points are labeled as *strong*, *good*, *weak*, *none*.
5. After specifying the ridge points, linking process is performed to fill the gaps.

3.2 Simulation of Skeleton Extraction Algorithm

Performance measure used to analyse the performance of the extended ridge point detection algorithm and Matlab simulation results are presented in this section. The output skeletons of different shapes are demonstrated in Figure 3.4 and 3.5.

Performance of the complete skeletonization process is measured using performance evaluation parameters which are defined in [45] and [26]. There are four main performance parameters: thinness measurement, connectivity measurement, sensitivity measurement and penetration measurement. This section explains these parameters briefly and gives the performance results of the algorithm based on the defined parameters.

Thinness Measurement (TM):

As described in Chapter 2, the skeleton should be as thin as possible. TM parameter measures the degree of thinness of the skeleton of an object in an image [45]. It counts triangles which can be constructed for each skeleton pixel T(S(i,j)) in 3x3 neighborhood as shown in Figure 3.6.

After calculating the triangle count for each skeleton point, total number of triangles (TM1) is specified using

$$TM1 = \sum_{i=1}^{m} \sum_{j=1}^{n} T(S(i,j)). \tag{3.6}$$

Finally, thinness measurement (TM) is expressed as:

$$TM = 1 - (TM1/TM2), \tag{3.7}$$

where TM2 shows the maximum number of triangles an image with size mxn could have and is computed by

$$TM2 = 8 * (max(m,n)). \tag{3.8}$$

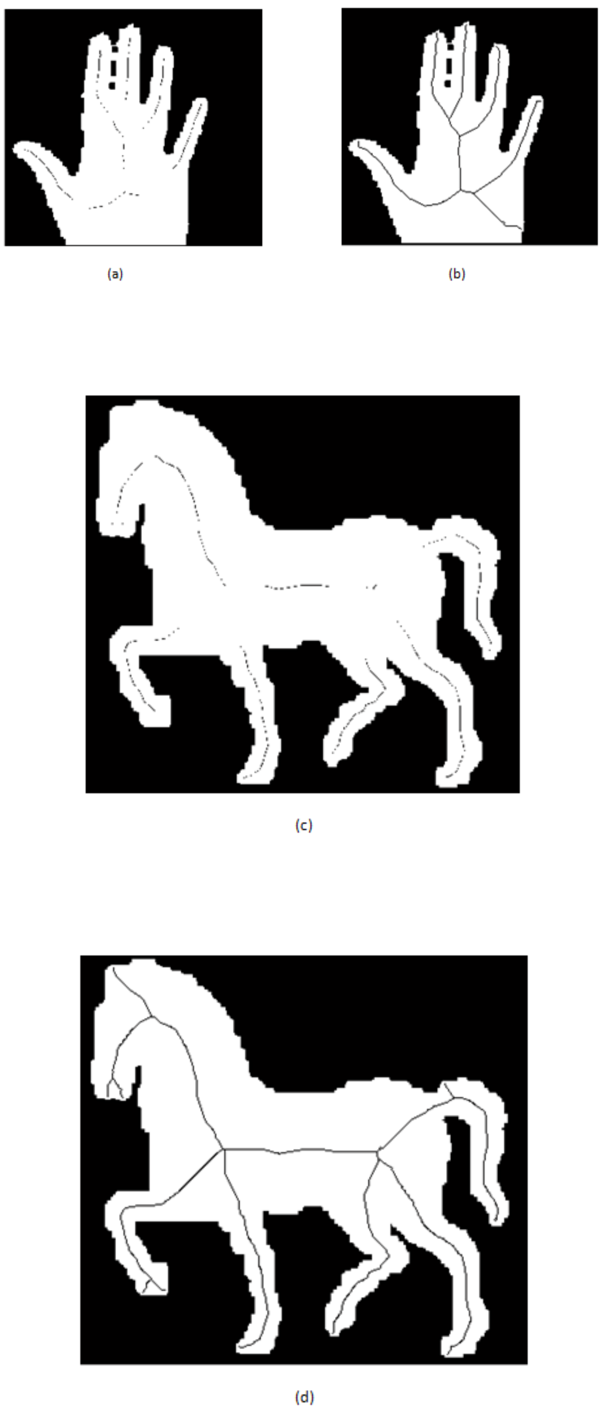


Figure 3.4: Test Results of Extended Algorithm: (a) Set of Ridge Points of a Hand (b) Skeleton Output of a Hand after Linking Process (c) Set of Ridge Points of a Horse (d) Skeleton Output of a Horse after Linking Process.

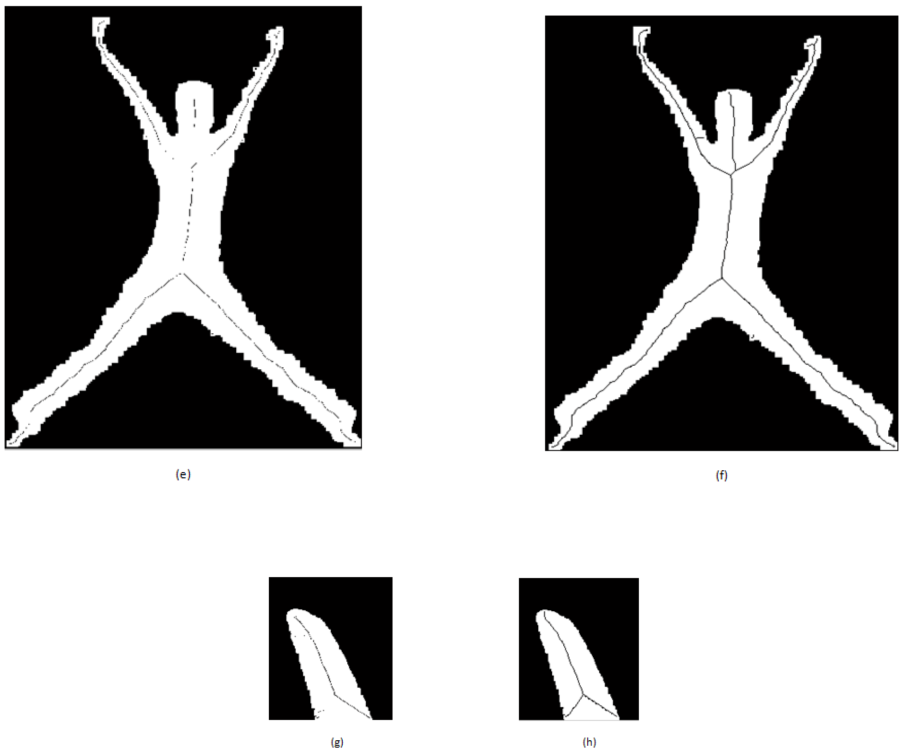


Figure 3.5: Test Results of Extended Algorithm(cont.): (e) Set of Ridge Points of a Man (f) Skeleton Output of a Man after Linking Process (g) Set of Ridge Points of a Finger (h) Skeleton Output of a Finger after Linking Process.

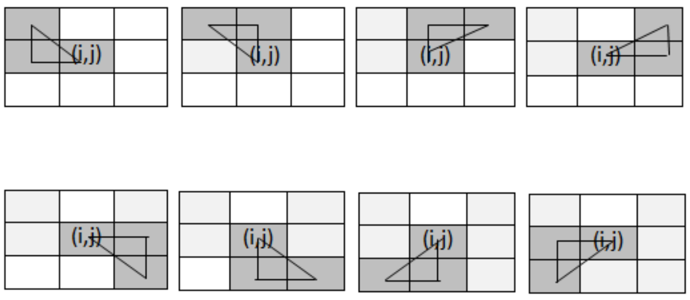


Figure 3.6: Possible Triangles on the Image

Connectivity Measurement (CM):

Another important parameter for performance evaluation is connectivity measurement. An object in an image is said to be disconnected if it has unconnected pieces [45]. Therefore, an active skeleton point which has less than two active neighbors in an 8-connectivity setting is said to be disconnected from the skeleton. The total number of points in this situation gives the connectivity information of the skeleton.

First, connectivity number for each skeleton point at position (i,j), CN(i,j), is calculated. Figure 3.7 displays the connectivity number for some example cases.

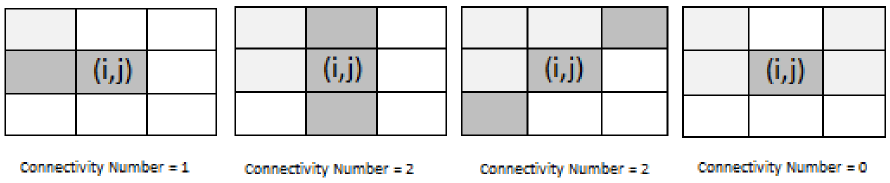


Figure 3.7: Connectivity Numbers

Psuedo code to calculate the connectivity number is printed in Figure 3.8.

```
for k=-1:1
  for l=-1:1
        if S(l+k,j+l)=1
           ConnectivityNumber(S(i,j)) = ConnectivityNumber(S(i,j))+1
  end
end
```

Figure 3.8: Psuedo Code to Calculate Connectivity Number

Consequently, connectivity measurement is stated as

$$CM = \sum_{i=1}^{m} \sum_{j=1}^{n} CM(i,j) \qquad \textbf{(3.9)}$$

where

$$CM(i,j) = \begin{cases} 1, & \text{if CN(i,j)} < 2 \\ 0, & \text{if CN(i,j)} > 2 \end{cases} \tag{3.10}$$

Sensitivity Measurement (SM):

Skeletonization techniques produce unwanted branches due to boundary noise. A good skeleton should not be have extra branches. SM measures the effects of the boundary noise to the skeleton.

An active skeleton point which has more than two active neighbors in an 8-connectivity setting is accepted as a branch point of the skeleton. The total number of these points give the sensitivity information of the skeleton.

The sensitivity measurement is defined as,

$$SM = \sum_{i=1}^{m} \sum_{j=1}^{n} SM(i,j), \tag{3.11}$$

where

$$SM(i,j) = \begin{cases} 1, & \text{if CN(i,j)} > 2 \\ 0, & \text{if CN(i,j)} < 2 \end{cases} \tag{3.12}$$

Penetration Measurement (PM):

PM counts the number of times that skeleton intersects the object boundary to measure the quality of skeleton S by means of topology preservation [45].

PM for the skeleton point at position (i,j) is stated as

$$PM(i,j) = \begin{cases} 1, & \text{if S(i,j)} \bigcap B \\ 0, & \text{else} \end{cases} \tag{3.13}$$

where B defines the boundary of the object.

PM for the skeleton is calculated using the equation

$$PM = \sum_{i=1}^{m} \sum_{j=1}^{n} PM(i,j). \tag{3.14}$$

The performance of the algorithm is evaluated on the images in Figure 3.8 using the specified parameters. The performance parameters are normalized by the image size to remove the ambiguity of image zoom or compression [45]. Table 3.1 denotes the measurement results. Higher thinness factor and lower connectivity and sensitivity factors are better skeletonization performance.

Table 3.1: Performance Measurement of Images.

Performance Parameter	TM	CM	SM	PM
Hand	0.9910	0.0088	0.0127	0.000103
Horse	0.9277	0.0082	0.0135	0.000745
Man	0.9626	0.0055	0.0059	0.000032
Finger	0.9486	0.0081	0.0093	0.000112

The performance parameters are also calculated under rotation for same images to measure the quality of the skeleton under rotations. The output skeletons of rotated images with different angles are shown in Figure 3.9 and the performance parameters can be found in Table 3.2. If Table 3.1 and 3.2 are compared for each image, it can be claimed that the performance of the skeletonization algorithm under rotation is very close to the original shape.

The presence of multiple objects in an image is automatically detected and the skeletons of these objects computed simultaneously. It is another advantage of the ridge detection algorithm.

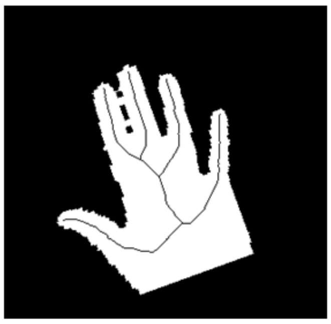
Table 3.2: Performance Measurement of Rotated Images.

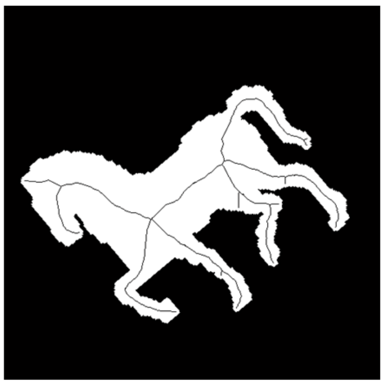
Performance Parameter	TM	CM	SM	PM
Hand	0.9763	0.0048	0.0076	0.000602
Horse	0.9649	0.0043	0.0058	0.000408
Man	0.9567	0.0058	0.0065	0.000531
Finger	0.9787	0.0065	0.0068	0.000166

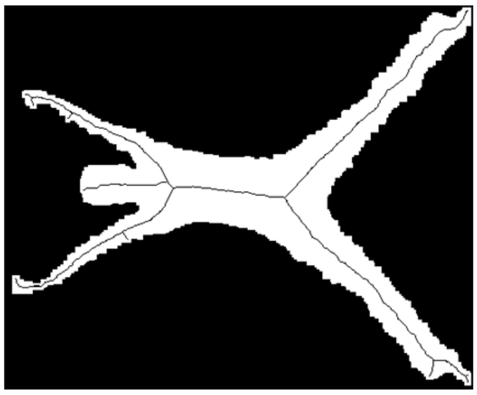
Table 3.3: Performance Measurement of the Algorithm in [1].

Performance Parameter	TM	CM	SM	PM
Hand	0.5726	0.0050	0.0072	0.000602
Horse	0.5905	0.0041	0.0069	0.000408
Man	0.4603	0.0052	0.0054	0.000531
Finger	0.7500	0.0045	0.0075	0.000166

In our algorithm, we benefit from both local maxima and possible ridges on the distance map. If we compare our algorithm with the techniques that use only local maxima for ridge point detection, it can be said that our algorithm acquires a higher thinning factor. Baja et al. [1] presents a technique that searches local maxima on


(a)


(b)


(c)

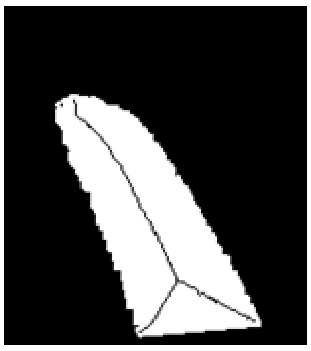

(d)

Figure 3.9: Test Results of Rotated Images : (a) Skeleton Output of a hand with a rotation of 20 Degrees (b) Skeleton Output of a horse with a rotation of 40 Degrees (f) Skeleton Output of a Man with a rotation of 90 Degrees (h) Skeleton Output of a Finger with a rotation 5 Degrees.

Chamfer 3-4 distance map. Baja's technique is also implemented in Matlab and the performance parameters are evaluated. The results can be examined in Table 3.3. It is obvious from the table that, this technique needs an additional thinning stage.

3.3 Detection and Classification of Skeleton Points for Hand Tracking

This section introduces the methodology of extracting branch points and end points from the skeleton, which is to be used in a hand tracking application.

Skeleton points are classified into three categories as mentioned in Chapter 2,

If a skeleton point has two active neighbors, it is assigned as a normal point. If a skeleton point has three or more active neighbors, it is assigned as a branch point. If a skeleton point has one active neighbor, it is assigned as an end point.

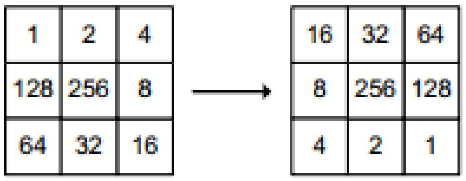


Figure 3.10: Filter Mask

A 2D convolution based approach is presented in [4] for classifying the skeleton points. The possible patterns, which are likely to generate an end point or a branch point, are determined a priori and these patterns are searched in the image by convolving the skeleton image with a single bidimensional filter. The filter mask that is used for feature extraction is shown in Figure 3.10 and possible patterns to search for end points and branch points are indicated with the output value of the convolution operation in Figure 3.11 and Figure 3.12 respectively .

The convolution is expressed as:

$$S'(x,y) = G(x,y) * S(x,y) \tag{3.15}$$

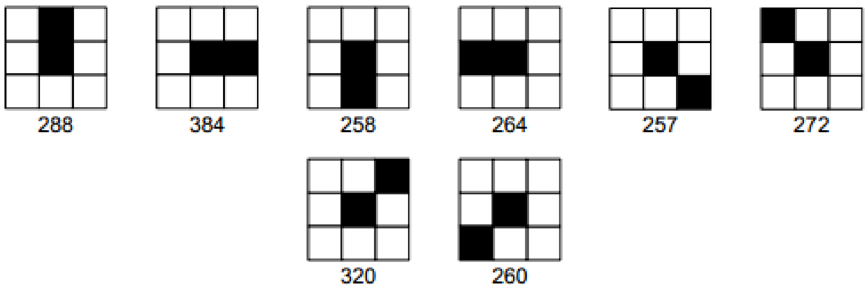


Figure 3.11: Possible End Points [4]

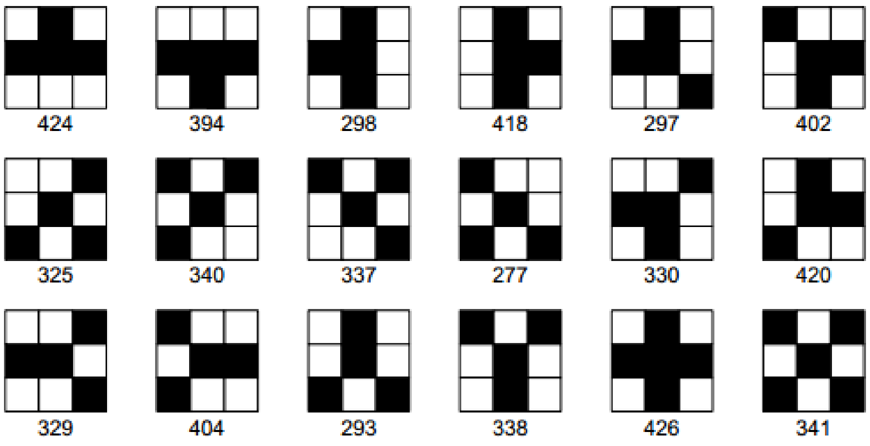
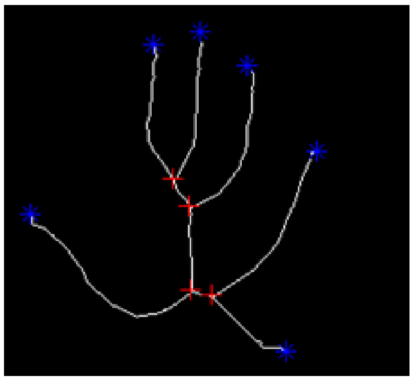


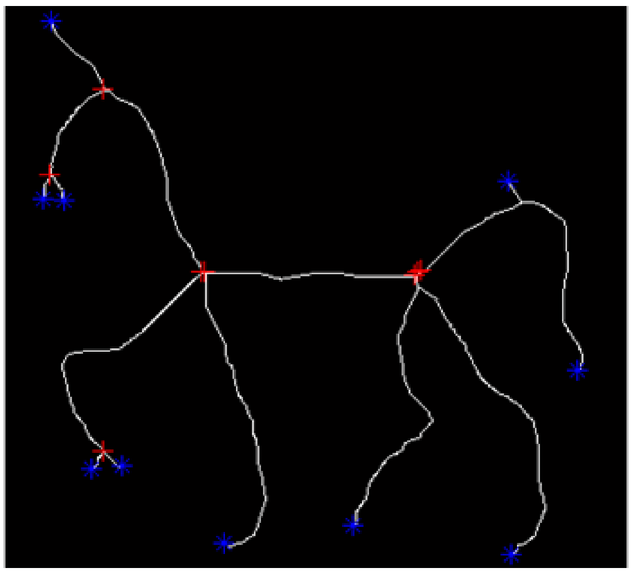
Figure 3.12: Possible Branch Points [4]

where S’(x,y) is the output of the convolution operation, G(x,y) is the bidimensional filter and S(x,y) is the skeleton. The output S’(x,y) is compared with possible end point and branch point patterns. If the output belongs to an end point pattern, it is labeled as an end point; if the output belongs to a branch point pattern, it is labeled as a branch point. Otherwise, it is labeled as a normal point.

This process is also implemented in Matlab. The detected branch points and end points of different images can be observed in Figure 3.13.



(a)



(b)

Figure 3.13: Branchpoints and Endpoints : a)Branch Points and End Points of a Hand b)Branch Points and End Points of a Horse

4. IMPLEMENTATION OF REAL TIME SKELETONIZATION SYSTEM

This chapter covers real time implementation of the skeletonization algorithm, which is described in Chapter 3. A fully pipelined skeleton exraction architecture is proposed for FPGAs. The design is tested with a hand tracking application using end points of the skeleton. All required steps for creating a complete skeletonization system for a hand tracking application are explained in detail. The whole system is developed on Digilent Atlys board with Spartan-6 FPGA. The frames are captured from CMOS image sensor with an integrated advanced camera system MT9D112. Figure 4.1 demonstrates the entire block diagram of the system on FPGA. The modules of the system can be listed as follows:

- Camera and DDR2 Control Module,
- Preprocessing Module,
- Background Subtraction and Morphological Cleaning Module,
- Distance Transform Calculation Module,
- Ridge Point Detection Module,
- Linking Module,
- DVI Transmitter Module,
- End points and Branch points Extaction Module,
- Hand Tracking Module.

The tracking applications that use skeletonization are preceded with object detection. Thus, background subtraction technique is performed to detect the objects as the first stage of the system. Morphological cleaning is an optional algorithm for such systems and it is performed to eliminate noise from the output binary images of the detection algorithm. Morphological cleaning is applied in this design since the noise can be misleading for the following modules. Once the object(s) is/are detected, the stages of the skeletonization system, which are listed in Chapter 3, are implemented. The first step of the algorithm is calculation of the distance map. Distance map is computed using Chamfer 3-4 metric after morphological cleaning. Ridge Point Detection Module realizes the second, third and fourth steps of the skeletonization algorithm. Recall that these steps are relative location calculation, local maxima determination and pattern search, respectively. Linking module implements the last stage of the skeleton extraction algorithm. It converts the set of ridge points to ridgelines to produce a connected skeleton. The output skeleton is produced by the linking module and is displayed on screen through DVI. End points and branch points are extracted as the next stage using the 2D convolution based method. Finally, Hand Tracking Module tracks the positions of end points throughout the frames captured by the integrated camera. The motion of the hand is displayed on leds of Atlys Board.

The following section give the implementation details of pipelined skeleton extraction architecture. The final section represents the experimental results of the full system.

4.1 Pipelined Skeleton Extraction Architecture on FPGA

This section introduces all the modules of the parallelized architecture. Finally, parallelism of the modules are explained on a timing diagram.

4.1.1 Camera and DDR2 control

This module contains two sub-modules: camera control and frame buffer control. Camera control sub-module configures an MT9D112 (which is a CMOS image sensor with an integrated advanced camera system) and provides a simple interface for reading the video data. Frame buffer controller operates as an interface for accessing a DDR2 external memory. It organizes read and write memory addressing. Moreover, this sub-module gives the opportunity to adjust frame capture rate and resize the video frame. This implementation captures video frames from MT9D112 camera using the camera controller and buffers them in DDR2 memory using frame buffer controller.

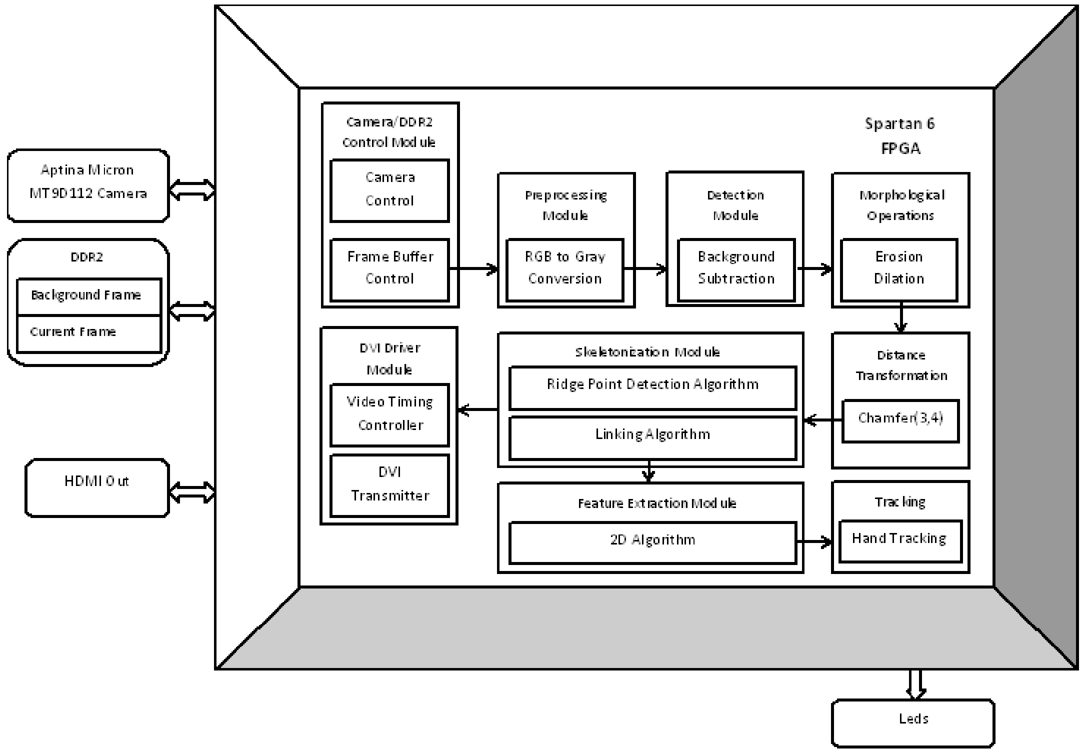


Figure 4.1: Architecture of FPGA-based Skeletonization System.

Also, the reference frame is stored in external memory to reduce the FPGA resource utilization.

4.1.2 Preprocessing

Camera controller module produces video frames in RGB format. Preprocessing module converts RGB images to gray scale images for further processing. Additionally, noise filtering is applied to gray scale images for the accuracy of the subsequent stages.

4.1.3 Object detection and morphological cleaning module

This module performs object detection and morphological cleaning algorithms. In present work, background subtraction method [46] is used for moving object detection. Background subtraction technique builds a representation of the scene and then finds deviations from the model for each incoming frame [46]. Representation of the scene is called background model. A common approach for specifying background model is to use information in a single frame. However, the background scene can also be built by averaging several successive frames to handle time varying background scenes. This task is called background training [47].

In our implementation, background training step takes a single frame and this frame is stored in external DDR2 RAM as a reference frame. Once the background model is obtained, classification of each pixel in the current frame is performed. The absolute difference between the current frame and the background model is used to specify objects [47]. If the difference value exceeds the threshold, the corresponding pixel is marked as foreground. Otherwise it is marked as background. The procedure for classification of the pixels is formulated as,

$$I_{cl} = \begin{cases} 1, & \text{if } |\mathrm{I_t} - \mathrm{I_{bg}}| > \text{threshold} \\ 0, & \text{if } |\mathrm{I_t} - \mathrm{I_{bg}}| < \text{threshold} \end{cases} \tag{4.1}$$

where I_t is the current frame, I_{bg} is the background frame and I_{cl} is the output frame.

After classification of the pixels in the current frame, morphological cleaning is applied using erosion and dilation filters with 3×3 structure element size. For the

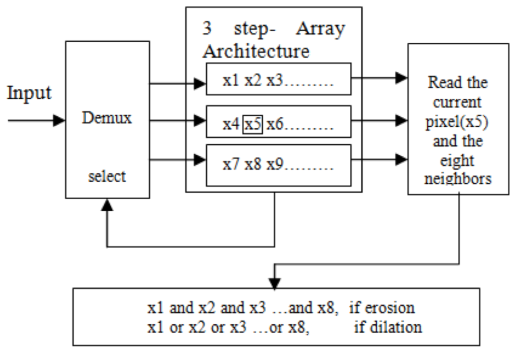

3 step- Array Architecture
Input
Demux
select
x1 x2 x3.........
x4 x5 x6.........
x7 x8 x9.........
Read the current pixel(x5) and the eight neighbors
x1 and x2 and x3 ...and x8, if erosion
x1 or x2 or x3 ...or x8, if dilation

Figure 4.2: Morphological Cleaning Block Diagram.

implementation of the morphological cleaning on FPGA, a pipelined array architecture sketched in Figure 4.2 is built. This architecture allows us to access all relevant pixels in parallel during morphological cleaning process. We use 3×3 structure element for morphological cleaning, thence three stage array architecture is utilized. Demultiplexer selects the proper array for the incoming new pixel. All the pixels in binary image are written to the relevant array in order. If eight neighbors of the relevant pixel are written to the array architecture, morphological operation is done for that pixel. The pipelined array architecture consumes only LUT resources of the FPGA. The size of the array architecture depends on the size of the image and the structure element.

4.1.4 Distance transform calculation

Distance transform calculation can be done either in parallel or sequentially [36]. Several sequential type [48] [49] and parallel type [50], [51] distance transform methods have been proposed in the literature. It is stated in [48] that the parallel approach consumes 16 times more resource than the sequential approach in [36]. So, sequential approach is preferred for FPGA implementation because of the low resource requirement.

The output of the morphological cleaning stage is the input for distance transform stage. Chamfer 3-4 metric is used for the hardware implementation of distance transform. The image is translated by propagating local distances first in backward direction and then in forward direction using backward and forward masks which are shown in Figure 4.3. The distance transform values are represented with 9 bits.

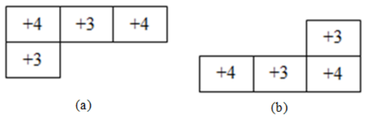


Figure 4.3: Distance Transform Forward and Backward Mask: (a) Forward Mask (b) Backward Mask.

The procedure of the forward and backward propagation is similar to each other. Infinite value is assigned to all foreground pixels at initial state and then backward mask is propagated on the whole image. The backward mask values are added to the relevant neighbors and the input pixel's distance value is compared with these neighbors. Then the pixel's distance value is updated to the minimum distance value of its neighbors. When the last pixel's distance value is assigned for the backward mask, forward propagation is carried out similarly using the forward mask. The hardware architecture of the backward and forward propagation is plotted in Figure 4.4. Shift Register Array (SRA) is used to store updated distance transform values. The size of the SRA is *W*x9 bits where *W* is the width of the input image.

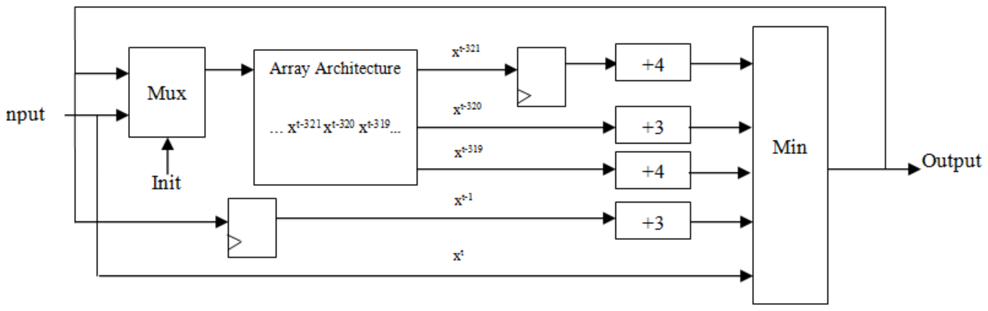


Figure 4.4: Distance Transform Backward and Forward Propagation Circuit.

The input pixel data of the distance transform comes from the morphological cleaning step in every clock pulse. In standard methodology of the sequential type distance transform, forward propagation step waits for all pixels to complete the backward propagation step. The backward propagation outputs are written to an internal block RAM and then forward mask is propagated on it. The distance transform calculation takes $W \times H \times 2$ clock cycles for a single frame with this standard type, where W and H are the width and height of the image respectively.

4.1.5 Ridge point detection module

Ridge Point Detection module takes the input pixels from the distance transformation module. When the forward propagation is completed for the first row of the image, skeleton extraction module starts the process instead of waiting for the completion of the backward propagation for all rows. The skeletonization algorithm is performed from right to left and bottom to top of the input image. The full skeletonization circuit is introduced in Figure 4.5. First of all, the relative locations of points (Nx and Ny) are calculated. The pixel D(i,j) and relevant neighbors D(i-1,j) and D(i,j-1) are read from the block RAM storing the propagation results of distance transform calculation. The final distance value of the pixel D(i,j) is produced after one clock cycle by the backward propagation circuit. Therefore, distance values of the neighbors are registered using D type flip flops. The comparator compares D(i,j) with the neighbors D(i-1,j) and D(i,j-1). The vectors Nx and Ny are designated according to the result of the comparator. Nx and Ny are represented with two bits for each point. "01" indicates that D(i,j) is greater than the relevant neighbor pixel, "10" indicates that D(i,j) is less than the relevant neighbor pixel and "00" indicates that D(i,j) is equal to the relevant neighbor pixel. Next step searches patterns on the image. For right to left scan on the image using Ny, the Ny values of previous column and next column on the same row are needed. So, the Ny values are registered using two stage flip flop as sketched in Figure 4.5. For the bottom to top scan on the image using Nx, the process is more complex. The Nx values of previous row and next row on the same column are needed. So, the consecutive three rows of the image are stored in three stage array architecture. Demultiplexer selects the proper array for the incoming new point. When the Nx values of relevant neighbors are written to the array as seen in Figure 4.5, the Nx values are sent to the pattern search block. Pattern search block looks for the -+, 0+,-0 and -0+ patterns. Also, it determines if the point is local maxima among eight neighbors and it specifies the maximum gradient path of each pixel for linking module. `pat_det_x` and `pat_det_y` signals indicate that if one of the patterns is

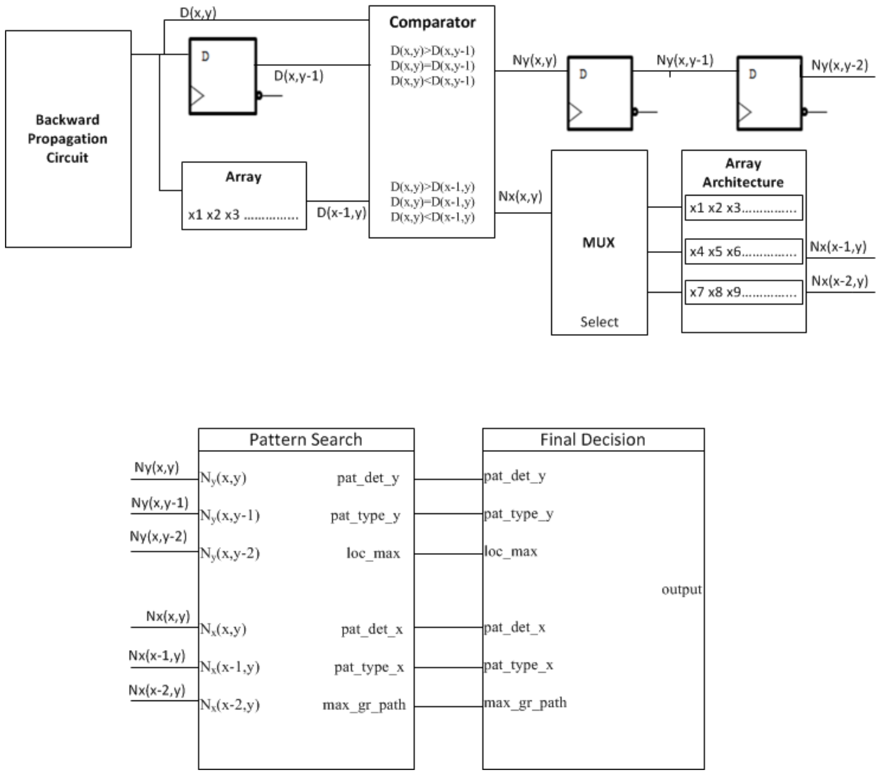


Figure 4.5: Circuit Implementation of Ridge Point Detection Algorithm.

detected on the scanline x and y respectively. `pat_type_x` and `pat_type_y` show the pattern type if detected. If a strong pattern is detected, it is pulled high. If a weak pattern is detected it is pulled low. `loc_max` denotes if the relevant point is local maxima and `max_gr_path` indicates the point with the maximum gradient to the relevant pixel. Decision block decides if the point is a skeleton point using `loc_max` , `pat_det_x`, `pat_det_y`, `pat_type_x` and `pat_type_y` according to rules that are described in Chapter 3.

4.1.6 Linking and feature extraction module

As mentioned in Chapter 3, gaps may occur due to discrete nature of distance transform, so a linking module is needed to convert set of ridge points to ridgelines.

2D convolution based pattern search approach, which is used for skeleton feature extraction, is also preferred for linking process to make it implementable on FPGA. The image is scanned left to right and the points that have one of the patterns within 3x3 neighborhood which are demonstrated in Figure 4.6 are re-examined. The gray

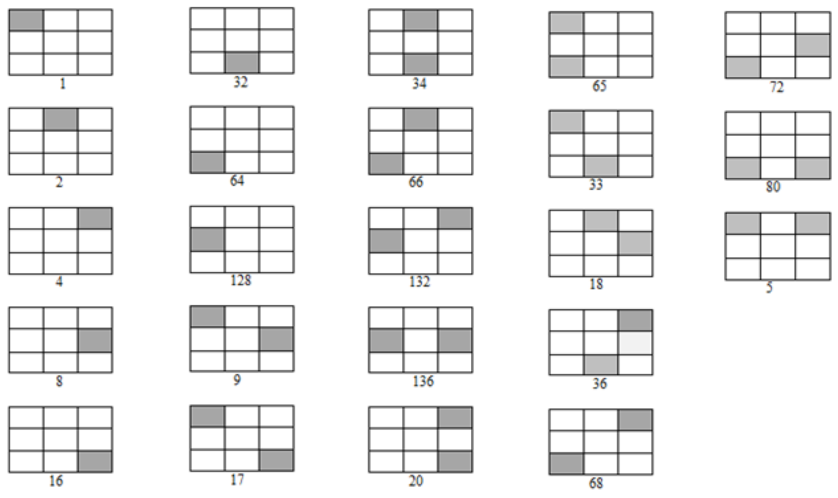


Figure 4.6: Linking Patterns.

pixels indicate the ridge points detected by ridge point detection algorithm. If the central point is a *weak* ridge or it is the maximum gradient one of the neighbors in the pattern, it is assigned as linking pixel. The same procedure is performed until all end points are connected to a branch point or a border point of the shape.

The implementation of 2D convolution on FPGA is illusrated in Figure 4.7. Skeleton image is convolved with the same bidimensional filter used in feature extraction process. Nine coefficients of the filter, which are expressed as a 3x3 matrix, are multiplied and added with consecutive three pixels in consecutive three rows as shown in Figure 4.7. Therefore, three consecutive rows must be stored in three stage array architecture. When two consecutive rows and three consecutive pixels of the last row are written to the array, the multiply and add process is started and the filtered pixel is generated. In this case, the skeleton image is defined with one bit, ’1’ or ’0’. Thus, multiply operation is realized using *AND* gates to reduce complexity. The convolution output is compared with the look up table that contains the pattern values in Figure 4.6.

Feature extraction module performs the same convolution operation to the output of the linking module to classify the branch points and end points. The filtered pixel is compared with the look up tables which hold the branch point and end point patterns that are declared in Chapter 3. If the filtered pixel is an end point, it is stored in the end point array; if it is a branch point, it is stored in the branch point array.

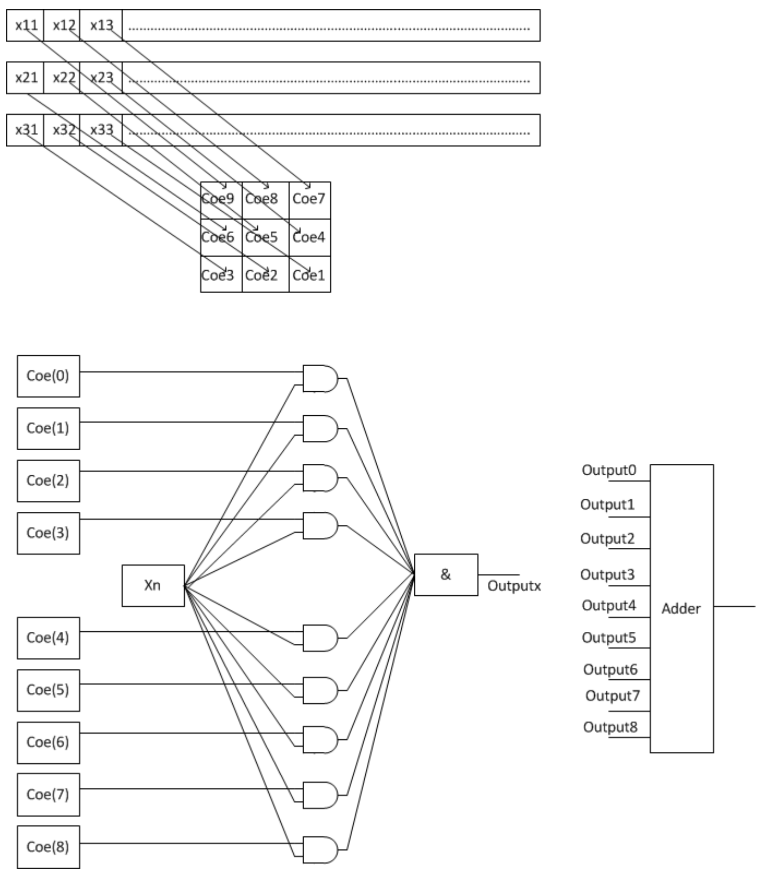


Figure 4.7: Circuit of 2D Convolution.

4.1.7 Hand tracking module

Hand tracking module takes the output of the feature extraction module as an input. The average of the endpoints (fingertips) of the hand is specified for each frame and the final decission is given comparing the average value with the average values of the following frames. Figure 4.8 demonstrates the circuit implementation of the hand tracking module.

4.1.8 DVI driver module

DVI driver module sends output images to HDMI port. It consists of two sub-modules: video timing controller and DVI transmitter. Video timing controller generates the proper synchronization signals according to the selected DVI resolution. In this work,

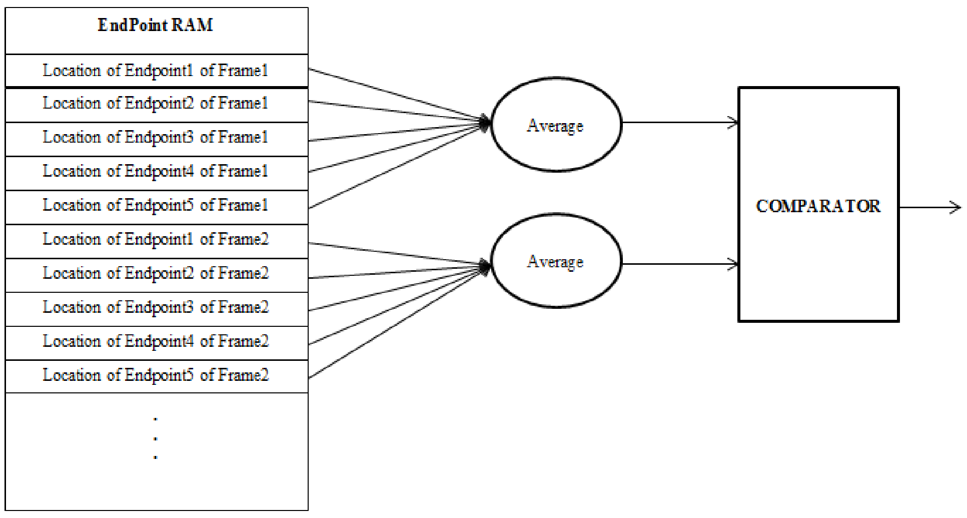

EndPoint RAM
Location of Endpoint1 of Frame1
Location of Endpoint2 of Frame1
Location of Endpoint3 of Frame1
Location of Endpoint4 of Frame1
Location of Endpoint5 of Frame1
Location of Endpoint1 of Frame2
Location of Endpoint2 of Frame2
Location of Endpoint3 of Frame2
Location of Endpoint4 of Frame2
Location of Endpoint5 of Frame2
Average
Average
COMPARATOR

Figure 4.8: Circuit Implementation of Hand Tracking Module.

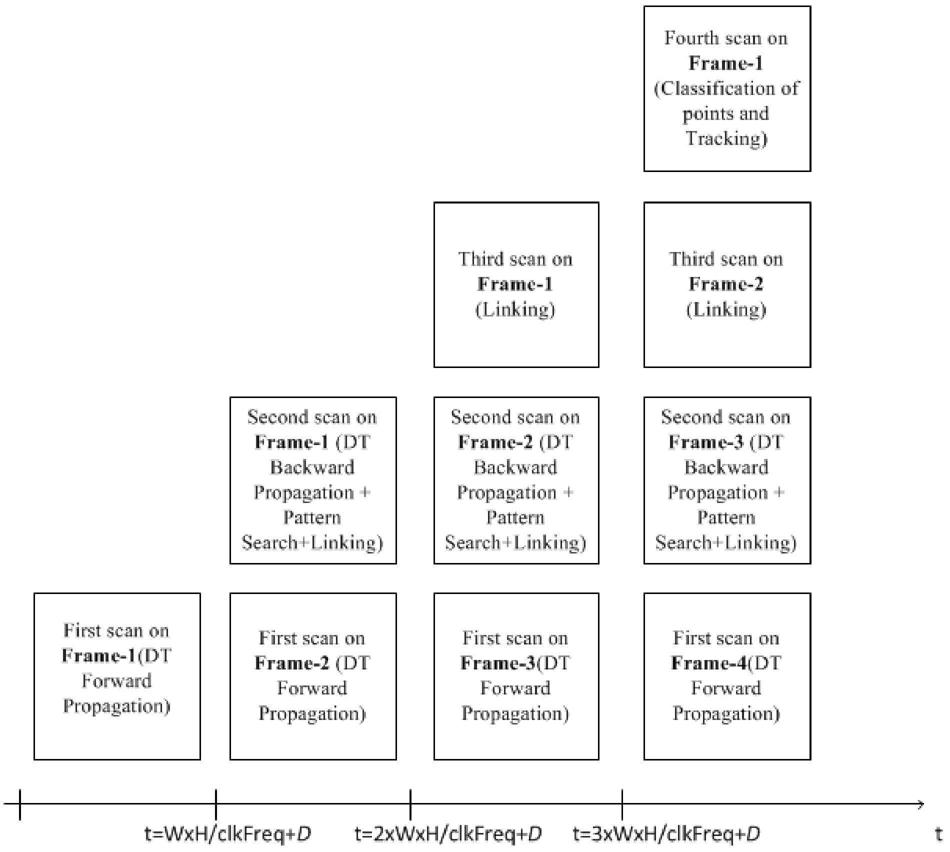

Fourth scan on Frame-1 (Classification of points and Tracking)
Third scan on Frame-1 (Linking)
Third scan on Frame-2 (Linking)
Second scan on Frame-1 (DT Backward Propagation + Pattern Search+Linking)
Second scan on Frame-2 (DT Backward Propagation + Pattern Search+Linking)
Second scan on Frame-3 (DT Backward Propagation + Pattern Search+Linking)
First scan on Frame-1(DT Forward Propagation)
First scan on Frame-2 (DT Forward Propagation)
First scan on Frame-3(DT Forward Propagation)
First scan on Frame-4(DT Forward Propagation)
t=WxH/clkFreq+D
t=2xWxH/clkFreq+D
t=3xWxH/clkFreq+D
t

Figure 4.9: Timing Diagram of Pipelined Architecture.

Table 4.1: FPGA Resource Usage.

Resource Type	Used
Slice Registers	2526
LUTs	2278
Block Memory	104

Table 4.2: Total Latency of the System.

Module	Latency(us)
Morphological Operations	17.8
Distance Transformations	1413.2
Skeleton Extraction	5652.8
Tracking	9.2
Total	7093

640x480 VGA resolution is used. DVI transmitter takes video data with proper sync signals and transmits them through a DVI/HDMI port. This application sends skeleton extraction outputs of the images to the HDMI port.

As observed from timing diagram in Figure 4.9, when a frame is scanned using forward mask for distance transformation, the backward mask is propagated for previous frame simultaneously. Thus, backward and forward propagation circuits work in parallel for consecutive frames. Ridge point detection and linking module process the data coming from the backward propagation circuit. It is not necessary to wait all the pixels to finish the bakward propagation. Only three consecutive rows must be finished. Therefore, these moduls are performed on the same scan as backward propagation. Feature extraction module also works in parallel with the ridge detection and linking module. As a result, a frame is processed in WxH/clkFreq. W is the width of the image, H is the height of the image. clkFreq is the operating clock frequency of the hardware.

4.2 Experimental Results

The proposed circuit implementation completes processing a single frame in WxHx6 clock cycles. W and H the width and height of the image respectively. In this work, the resolution of the video frames is 320x480. The Digilent Atlys board is used for the hardware implementation. The whole system is operated at 108 MHz clock frequency.

The resource usage for the FPGA and total latency of the system are shown in Table 4.1 and Table 4.2 respectively.

Also, figure 4.10 demonstrates the full system with FPGA board and camera module. Some illustrations of the real time system are shown in 4.10.

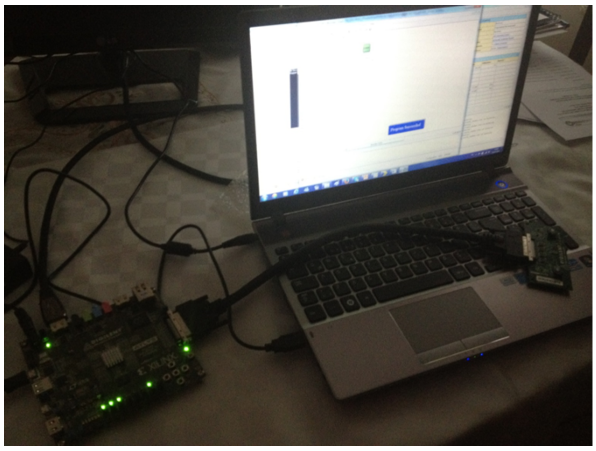


Figure 4.10: Full System with FPGA Board and Camera Module.

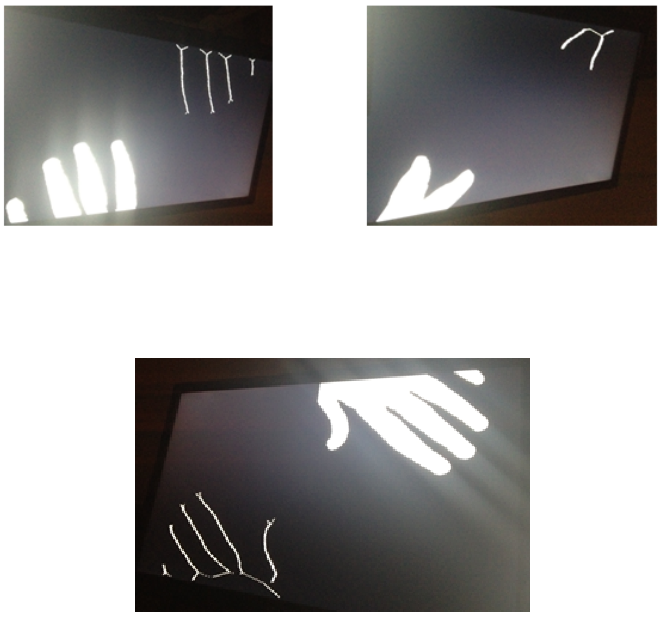


Figure 4.11: Some Illustrations of the Real Time System.

5. CONCLUSIONS AND RECOMMENDATIONS

In this thesis, a real time skeletonization system is presented for implementation on FPGA. The performance of the system is observed on a hand tracking application for remote control of television sets.

Skeletonization is exploited in many image processing systems which require real time processing capabilities. However, skeleton extraction methods can be computationally complex which makes it impossible for the system to cope with the real-time requirements. For our real time implementation, a simple but an efficient distance transform based skeleton extraction algorithm is elected [31]. This algorithm is very suitable for VLSI implementation; but boundary noise effects are observed in the output skeleton. In this respect, an extended ridge detection algorithm is proposed based on [1] to reduce the corruptive effects of noise. The extended ridge detection algorithm is compared with the algorithm in [1] and it is observed that the proposed method produces thinner skeleton.

A fully parallelized skeleton extraction architecture is proposed for the real time implementation of the extended algorithm on FPGA. The complete system is developed on Digilent Atlys board with Spartan-6 FPGA to test the suggested skeletonization architecture. The frames are captured from CMOS image sensor with an integrated advanced camera system MT9D112. Whole image is processed in 6xWXH clock cycles. W represents width and H represents the height of the frame. In this work, the size of the frames is 320x480 and the operating frequency of the complete circuit is 108 MHz. In other words, this system can process 110 frames per second.

Performance of the novel algorithm is evaluated according to the widely acknowledged performance measures for skeletonization research such as thinness, sensitivity, connectivity and penetration. Resource utilization and timing performance of the

FPGA implementation are investigated for comparison with similar systems in literature.

5.1 Future Work

The detection and segmentation of the moving object is a critical step for tracking systems. A poor segmentation makes the whole system fail. On the other hand, some works implement complex algorithms which are not convenient for real time applications. In this work, a simple version of background subtraction technique is used for object detection since object detection is out of scope of the thesis. Replacing the detection stage with an algorithm, which is more efficient as well as suitable for hardware implementation, assists the proposed skeletonization system for a better performance.

REFERENCES

[1] **Sanniti di Baja, G.**, 2003. Well-shaped, stable, and reversible skeletons from the (3,4)-distance transform, *J Visual Commun Image Represent*, **5(1)**, 107–115.

[2] **Palágyi, K.**, `http://www.inf.u-szeged.hu/~palagyi/skel/skel.html`, date retreived: 09.10.2013.

[3] **Blum, H.**, 1967. A Transformation for Extracting New Descriptors of Shape, **W. Wathen-Dunn**, editor, Models for the Perception of Speech and Visual Form, MIT Press, Cambridge, pp.362–380.

[4] **Olsen, M.**, **Hartung, D.**, **Busch, C. and Larsen, R.**, 2011. Convolution approach for feature detection in topological skeletons obtained from vascular patterns, Computational Intelligence in Biometrics and Identity Management (CIBIM), 2011 IEEE Workshop on, pp.163–167.

[5] **Siddiqi, K.**, **Shokoufandeh, A.**, **Dickenson, S. and Zucker, S.**, 1998. Shock graphs and shape matching, Computer Vision, 1998. Sixth International Conference on, pp.222–229.

[6] **Ruberto, C.D.**, 2004. Recognition of shapes by attributed skeletal graphs, *Pattern Recognition*, **37**, 21–31.

[7] **Hancock, T.E.R.**, 2004. A skeletal measure of 2D shape similarity, *Computer Vision and Image Understanding*, **95**, 1–29.

[8] **Zhu, S. and Yuille, A.**, 1995. FORMS: a flexible object recognition and modelling system, Computer Vision, 1995. Proceedings., Fifth International Conference on, pp.465–472.

[9] **Geiger, D.**, **Liu, T.L. and Kohn, R.**, 2003. Representation and self-similarity of shapes, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **25(1)**, 86–99.

[10] **Asian, C. and Tari, S.**, 2005. An axis-based representation for recognition, Computer Vision, 2005. ICCV 2005. Tenth IEEE International Conference on, volume 2, pp.1339–1346 Vol. 2.

[11] **Sebastian, T.**, **Klein, P. and Kimia, B.**, 2004. Recognition of shapes by editing their shock graphs, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **26(5)**, 550–571.

[12] **Chen, C. and Liu, S.**, 2012. Detection and Segmentation of Occluded Vehicles Based on Skeleton Features, Instrumentation, Measurement, Computer, Communication and Control (IMCCC), 2012 Second International Conference on, pp.1055–1059.

[13] **Yogameena, B.**, **Mansoor Roomi, S.**, **Jyothi Priya, R.**, **Raju, S. and Abhaikumar, V.**, 2012. People/vehicle classification by recurrent motion of skeleton features, *Computer Vision, IET*, **6(5)**, 442–450.

[14] **Jimenez, P.G.**, **Lopez, B.**, **Cueco, R.T.**, **Campilho, A. and Sastre, R.**, 2012. Hand Detection and Tracking Using the Skeleton of the Blob for Medial Rehabilitation Applications, Proceedings of the 9th international conference on Image Analysis and Recognition, volume 3, pp.130–137.

[15] **Fang, B.**, **Wen, H.**, **Liu, R.Z. and Tang, Y.Y.**, 2010. A New Fingerprint Thinning Algorithm, Pattern Recognition (CCPR), 2010 Chinese Conference on, pp.1–4.

[16] **Gang, C.**, **Ning, C. and Yong, Z.**, 2012. An improved OPTA fingerprint thinning algorithm based on neighborhood searching, Computer Science and Information Processing (CSIP), 2012 International Conference on, pp.637–640.

[17] **Saleh, A.**, **Eldin, A. and Wahdan, A.M.**, 2009. A modified thinning algorithm for fingerprint identification systems, Computer Engineering Systems, 2009. ICCES 2009. International Conference on, pp.371–376.

[18] **Xu, D.**, **Li, B. and Nijholt, A.**, 2009. A Novel Approach Based on PCNNs Template for Fingerprint Image Thinning, Computer and Information Science, 2009. ICIS 2009. Eighth IEEE/ACIS International Conference on, pp.115–119.

[19] **Palágyi, K. and Kuba, A.**, 1998. A Hybrid Thinning Algorithm for 3D Medical Images, *Computing and Information Technology*, **6**, 149–164.

[20] **Palágyi, K.**, **Sorantin, E.**, **Balogh, E.**, **Kuba, A.**, **Halmai, C.**, **Erdôhelyi, B. and Hausegger, K.**, 2001. A Sequential 3D Thinning Algorithm and Its Medical Applications, *Int. Conf. Information Processing in Medical Imaging, IPMI.*

[21] **Bourbakis, N.**, **Steffensen, N. and Saha, B.**, 1997. Design of an array processor for parallel skeletonization of images, *Circuits and Systems II: Analog and Digital Signal Processing, IEEE Transactions on*, **44(4)**, 284–298.

[22] **Lopich, A. and Dudek, P.**, 2005. Architecture of asynchronous cellular processor array for image skeletonization, Circuit Theory and Design, 2005. Proceedings of the 2005 European Conference on, volume 3, pp.III/81–III/84 vol. 3.

[23] **Gayathri, S. and Sridhar, V.**, 2013. An Improved Fast Thinning Algorithm for Fingerprint Image, *International Journal of Engineering Science and Innovative Technology, IJESIT*, **2(1)**, 264–270.

[24] **Xu, H.**, **Qu, Y.**, **Zhang, Y. and Zhao, F.**, 2009. FPGA Based Parallel Thinning for Binary Fingerprint Image, Pattern Recognition, 2009. CCPR 2009. Chinese Conference on, pp.1–4.

[25] **Hermanto, L.**, **Sudiro, S. and Wibowo, E.**, 2010. Hardware implementation of fingerprint image thinning algorithm in FPGA device, Networking and Information Technology (ICNIT), 2010 International Conference on, pp.187–191.

[26] **Lakshmi, J.K. and Punithavalli, M.**, 2009. A Survey on Skeletons in Digital Image Processing, Proceedings of the International Conference on Digital Image Processing, ICDIP '09, IEEE Computer Society, Washington, DC, USA, pp.260–269, `http://dx.doi.org/10.1109/ICDIP.2009.21`.

[27] **Ranganathan, N. and Doreswamy, K.B.**, 1995. A VLSI chip for computing the medial axis transform of an image, Computer Architectures for Machine Perception, 1995. Proceedings. CAMP '95, pp.36–43.

[28] **Siddiqi, K. and Pizer, S.**, 2009. Medial Representations: Mathematics, Algorithms, Applications, Springer.

[29] **N., S.**, 2003. Design of a cellular architecture for fast computation of the skeleton, *VLSI Signal Processing*, **35(4)**, 61–73.

[30] **Hajdu, A.**, **Hajdu, L. and Tijdeman, R.** Approximation of Euclidean distances by chamferdistances, *Unpublished*.

[31] **Chang, S.**, 2007. Exracting Skeletons from Distance Maps, *International Journal of Computer Science and Network Security*, **7(7)**, 213–219.

[32] **Hardenberg, C. and Bérard, F.**, 2001. Bare-Hand Human-Computer Interaction, Proceedings of the ACM Workshop on Perceptive User Interfaces, pp.1–8.

[33] **Tagliasacchi, A.**, 2013. Skeleton Extraction and Skeleton-Driven Processing of Incomplete Data, Ph.D. thesis.

[34] **Mather, J.N.**, 1983. Distance from a sub-manifold in Euclidean Space, In proceedings of Symposia in Pure Mathematics.

[35] **Bai, X.**, **Latecki, L. and yu Liu, W.**, 2007. Skeleton Pruning by Contour Partitioning with Discrete Curve Evolution, *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, **29(3)**, 449–462.

[36] **Borgefors, G.**, 1986. Distance Transformations in Digital Images, *Computer Vision, Graphics, and Image Processing*, **34(3)**, 344 – 371.

[37] **Smith, M.J.**, 2004. Distance and Path, Ph.D. thesis.

[38] **Choi, W.P.**, **Lam, K.M. and Siu, W.C.**, 2003. Extraction of the Euclidean Skeleton Based on a Connectivity Criterion, *Pattern Recognition*, **36**, 721–729.

[39] **L. J. Latecki, Q. Li, X.B. and Liu, W.**, 2007. Skeletonization Using SSM of the Distance Transform, International Conference on Image Processing, ICIP, volume 5, pp.349–352.

[40] **Url-1**, `http://en.wikipedia.org/wiki/Voronoi_diagram#Formal_definition`, date retreived: 15.10.2013.

[41] **Lohman, G.**, 1998. Volumetric Image Analysis, Ph.D. thesis.

[42] **Kresch, R. and Malah, D.**, 1998. Skeleton-based morphological coding of binary images, *Image Processing, IEEE Transactions on*, **7(10)**, 1387–1399.

[43] **Curic, V.** Mathematical Morphology and Distance Transforms, *Uppsala University*.

[44] **Dimitrov, P.**, **Phillips, C. and Siddiqi, K.**, 2000. Robust and efficient skeletal graphs, Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on, volume 1, pp.417–423 vol.1.

[45] **Mann, N. and P., S.**, 2012. Medial Axis Transformation based Skeletonzation of Image Patterns using Image Processing Techniques, *International Journal of Science and Research, IJSR*, **1(3)**, 220–223.

[46] **A. Yilmaz, O.J. and Shah, M.**, 2006. Object Tracking : A Survey, *ACM Computing Surveys (CSUR)*, **38(4)**.

[47] **S. Liu, A. Papakonstantinou, H.W.D.C.**, 2011. Real-Time Object Tracking System on FPGAs, Application Accelerators in High-Performance Computing (SAAHPC), 2011 Symposium on, pp.1–7.

[48] **S. Hezel, A. Kugel, R.M. and Gavrila, D.**, 2002. FPGA- based Template Matching using Distance Transforms, Field-Programmable Custom Computing Machines, 2002. Proceedings. 10th Annual IEEE Symposium on, pp.89–97.

[49] **J. Maurer, R. Calvin, R.Q.V.R.**, 2003. A linear time algorithm for computing exact Euclidean distance transforms of binary images in arbitrary dimensions, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **25**, 265–270.

[50] **Sudha, N.**, 2005. A pipelined array architecture for Euclidian distance transformation and its FPGA implementation, *Microprocessors and Microsystems*, **29**, 405–410.

[51] **Yu-Hua Lee, Shi-Jinn Horng, J.S.**, 2003. Parallel computation of the Euclidean distance transform on a three-dimensional image array, *IEEE Transactions on Parallel and Distributed Systems*, **14**, 203–212.

PHOTO

CURRICULUM VITAE

Name Surname: Melike Atay

Place and Date of Birth: Beyoğlu, 1988

E-Mail: melike.atay@gmail.com

B.Sc.: Yıldız Technical University

List of Publications and Patents:

PUBLICATIONS/PRESENTATIONS ON THE THESIS

- Atay, M. and Yalcin, Mustak E., 2013: A parallelized distance transformation architecture for FPGAs *Circuit Theory and Design (ECCTD), 2013 European Conference on*, September 8-12, 2013 Dresden, Germany.