REAL TIME MOTION CAPTURE FOR VR APPLICATIONS

Sarosij Bose

sarosij.bose@smu.ca

MITACS GRI Summer Intern

Graphics and Spatial Computing (GSC) Lab

20th June – 15th September, 2022

UNDER THE GUIDANCE OF

Dr. Jiju Poovvancheri

Assistant Professor

Department of Math and Computing Science

Saint Mary's University

Submitted on: 12th September, 2022.

Contents

List of Figures

Acknowledgements

This work is a part of the MITACS Globalink Summer Internship Project which was assigned to me under the guidance of Dr. Jiju Poovvancheri at Saint Mary's University, Halifax, Nova Scotia, Canada. The aim of this project was to develop a full body tracking algorithm which can be used for realtime applications such as in AR/VR based systems.

I really want to express my gratitude to Dr. Poovvancheri for selecting me for this project which enabled me to work on a novel and different aspect of Computer Graphics and explore a different domain outside my expertise. This domain being new to me, the way he guided and supported me in every sense from detailed discussions, meetings over various concepts to clearing doubts, is beyond words to express. It also helped me understand the basics of graphics and vision to a much more fundamental level and think in a novel manner.

Further, this opportunity enabled me to develop my research skills, ethics and culture. Combining and applying the concepts and experience from my previous research experiences and college courses has been the most enthralling and eye opening part in this project for me. Applying a completely new graphics fundamental structure, using it to develop a novel algorithm using very different libraries, languages and frameworks has been my biggest learning so far in this project.

Finally, I express my sincere gratitude to all the lab members of the Graphics and Spatial Computing (GSC) Lab, for helping me settle down during my initial days, to showing me how everything at the lab works and for being by my side whenever required. I am also thankful to SMU Maths and Computing Department for providing me with all the required Lab facilities at McNally North Building and ensuring everything was in order.

1 Introduction

Abstract. We present a novel shape approximation method using a pill decomposition approach given the surface points and their corresponding normals at each point on the surface. We first extract the maximal empty sphere representation of a given input shape and then construct the 'pill': consisting of two sphere meshes. These collection of pills are progressively decomposed to obtain a good approximation of the original shape. Our algorithm is easy to reuse and implement and is currently available in a multi-processing setup. To ensure reproducibility and further research, the source code and raw data has also been released.

Modern devices such as Virtual Reality (VR) need robust real time body tracking systems to track the particular objects which may come in front of it. This means there is a need of an **appropriate representation**, which when combined with a **lightweight customized tracking algorithm** can be readily deployed with ease on an **edge device**.

In this report, we introduce a modified geometric representation mesh structure known as the 'pill'. We propose a new algorithm based on this representation to approximate the shapes of various objects using the Medial Axis Transform. Further, we also perform extensive analysis on a variety of shapes and sizes using our 'pill decomposition approach' and obtain good approximations as reported in the sections below.

2 Methodology

2.1 Geometric Representations

The fundamental unit or shape using which a particular 3D structure is represented is known as the geometric representation for that particular shape.

Following are some commonly used geometric representations.

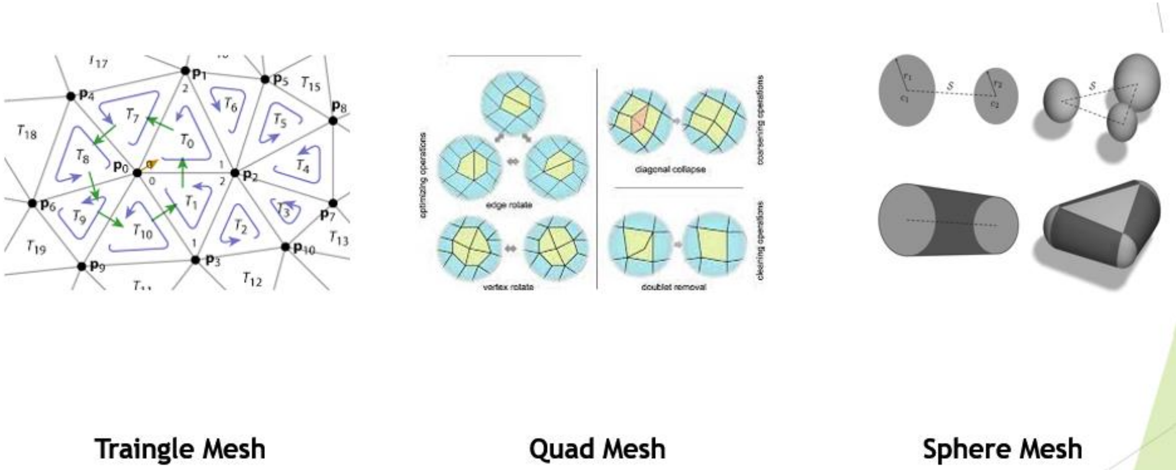

Figure 2.1(a)

In this report, we employ 'sphere meshes' to represent the full body of a human. It was shown in [1] that sphere meshes were the most efficient complexity wise for dealing with human like shapes. Therefore, we considered sphere meshes to be the most appropriate representation in this case.

2.2 Existing Approaches

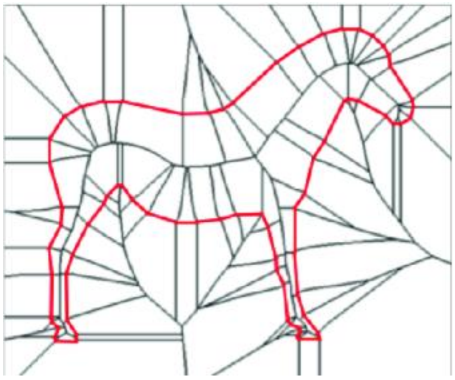

Figure 2.2(a): Voronoi diagram/Ray tracing based methods

The idea is to construct an approximate represenatation using a basic unit (covered below). Such methods are less prone to discretization errors leading to higher precision but they require heavy computational requirements.

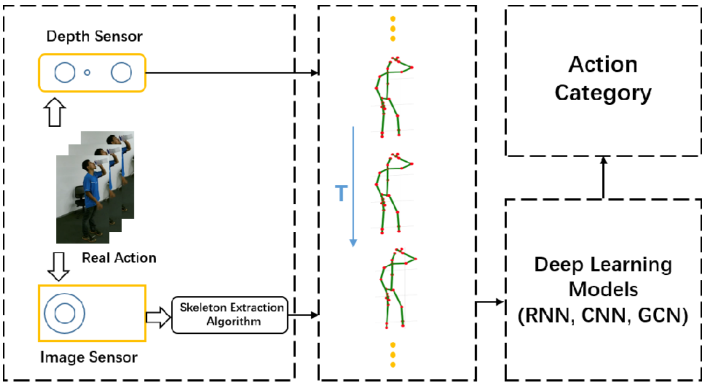

Figure 2.2(b): Deep Networks

Deep Learning based networks extracts the features (ex: skeleton points) from a certain Region of Interest (ROI) and then uses a GCN (or CNN in some cases) to obtain the frame level prediction. This is then extrapolated over the previous/next frames if necessary.

2.3 Tracking Techniques

Tracking techniques can be broadly divided into 2 categories:-

- Generative Tracking
- Discriminative Tracking

Each of these approaches have their own respectively advantages and disadvantages:-

- Discriminative tracking procedures are usually more precise (i.e; prevent error propagation) and they need more data.
- Generative tracking procedures can be re-initialized and they are based on templates. However, final convergence and optimization depends significantly on the initial template chosen.

An ideal scenario would usually be a mix of the two methods: Develop an initial novel representation (generative) and use it for realtime (discriminative) hand tracking.

2.4 Sphere Meshes

Why Sphere Mesh?

- In [1], it was shown that a sphere mesh based tracking template performed significantly better than other template based models. Some of it's advantages were:-

  * Complex motion tracking

  * Tracking at 60 FPS

  * No need of per-frame re-initialization.

An additional advantage of using sphere meshes is that it's exceptionally stable – usually MAT based algorithms are known to be notoriously hard to properly optimize to a particular shape's boundary [2]. This in turn helped to design our proposed algorithm efficiently.

2.5 Medial Axis Transform (MAT)

What does Medial Axis Transform (MAT) actually mean?

- A medial ball is a ball that fits completely inside B and does not contain any other ball that would fit inside B. The MAT is defined as the set of points that are the centres of all medial balls of B (see Figure in the previous slide). Each medial ball touches the medial ball B in at least two points, called its feature points.

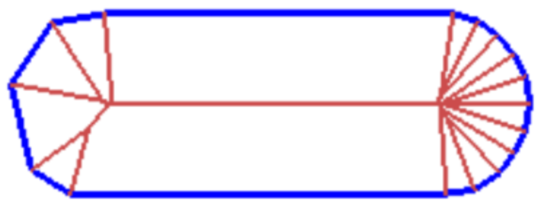

Figure 2.5(a): MAT for a random shape

Why is it useful?

The MAT can be typically subdivided into open clusters that correspond to various features such as curves and sharp edges in the given shape. Further, such representations expose different propertiesof a shape which has their own unique use cases. This MAT can then be used for further post-processing where it can be used to extract specific features.

2.6 Rotating Calipers (R.C.) Algorithm

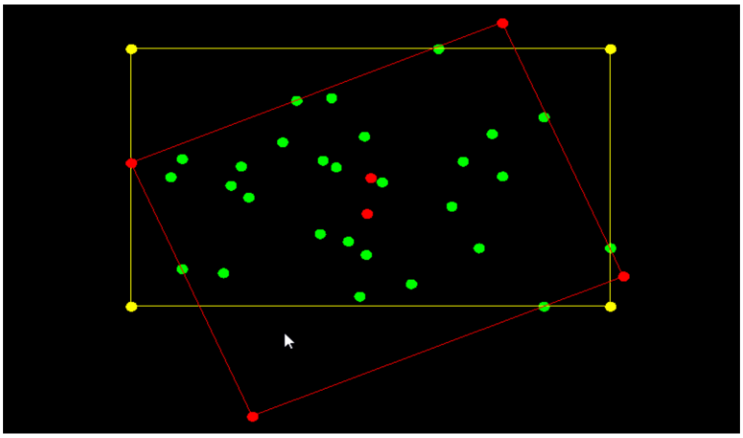

Figure 2.6(a): Convex Hull for a set of points. Points in red are antipodal.

This algorithm is used to find out the set of antipodal points in a convex 2D hull. This algorithm has a complexity of O(NLogN). This is especially useful in implementing our algorithm as we are required to find out the furthest point in the surface from the initial chosen point.

2.7 Initial Ball and Point Construction

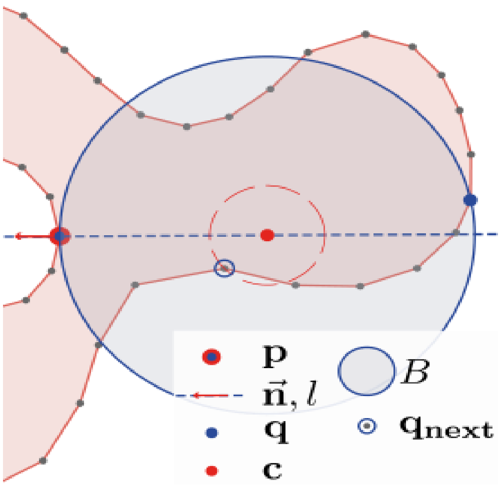

Figure 2.7(a): Initial Ball construction

Initial Ball construction

We use the algorithm proposed in [4] to obtain the initial maximal ball. The initial ball radius is set which is iteratively made to be maximal by running a KDTree query to find out the nearest neighbours from that point and checking whether the radius equals the distance from the sample point to the center of the ball under consideration.

Initial point

This can be any random point from the given set of coordinates. We employ a simple randomizer to pick any index from a given input list of points.

2.8 Iterative Radius

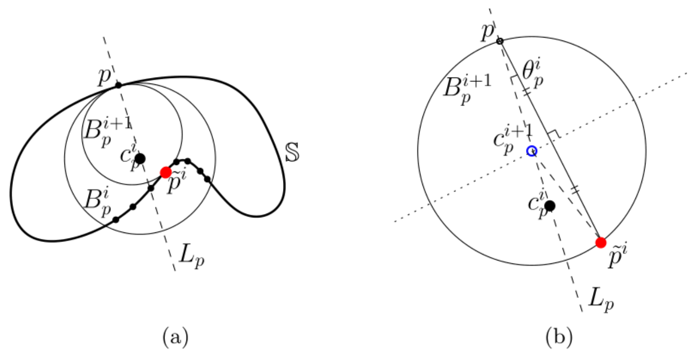

Figure 2.8(a): Iterative refinement of the radius at each step

The radius for each maximal empty ball is calculated iteratively by applying the simple formula below:-

$$\rho_p^{i+1} = \frac{d(p, \tilde{p}^i)}{2\cos\theta_p^i},$$

Each of these radiuses obtained has to satisfy certain criterions:-

- **If r<0:** Then the ball has been computed on the other side of the curve and hence can't be taken into consideration
- **If r>init radius:** Otherwise the algorithm will diverge.

3 Experiments and Results

3.1 Pill: A New Representation

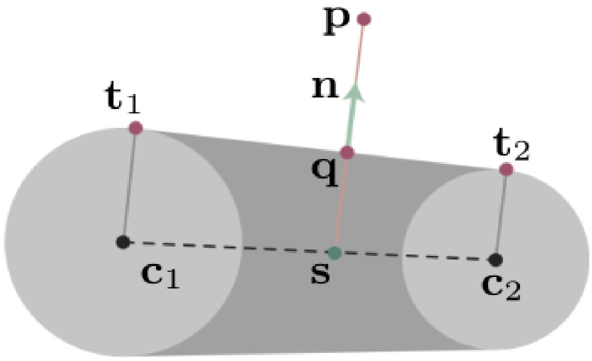

Figure 3.1(a): The Pill structure

Pill: The 'Pill' data structure can be thought of as a set of 2 sphere mesh balls which are connected by 2 direct common tangents (DCTs). It can be thought of as a 'primitive' building unit.

Applications: This construct is particularly important for approximating surfaces – these can be thought of as a fundamental representation which can be further broken down as far as possible. The more these 'pills' are broken down, the better the obtained figure is for that shape.

3.2 Proposed Algorithm

Our proposed algorithm has been outlined below.

Algorithm 1: Approximation to a particular provided shape

Input: Initial radius r, downsample d, neighbours k, iterations n

Output: Approximated shape for the provided input

Data: args, Shape: Point and Normal Information

```
pill_info = {}
ball_info = {}
ball_count = pill_count = 0
data ← Downsample(data, d)
for iter in range(n) do
    if iter == 0 then
        bar = 1
    bar = length(pill_info)
    for pill_id in bar do
        pill = pillConstruct(ball_info,
          idx1=parent_pill[ball1],
          idx2=furthest_ball_info)
        pill_count = pill_count + 1 furthest_pt =
          project_furthestpt(data, tangent1, tangent2)
        max_ball = MASB(args,
          max_r = r, k = k, furthest_pt)
        centre, radius = max_ball.compute_ball()
          ball_info[ball_count] = centre, radius
          ball_count = ball_count + 1
    furthest_ball_info = ball_count
    parent_pill = pill_info[pill_count]
Get ball_info, pill_info
```

Here, MASB refers to the maximal empty ball class.

3.3 Maximal Empty Ball figures

Below are the obtained Maximal empty Ball figures for shapes are shown:-

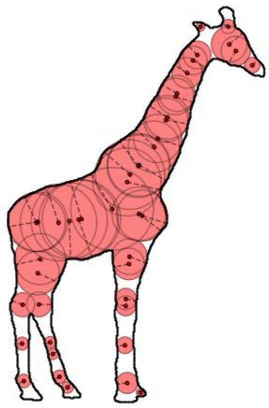

Figure 3.3(a): A giraffe

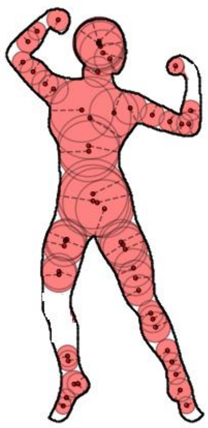

Figure 3.3(b): A Human

3.4 Approximated Figures

Some obtained shape approximations for some figures are shown below. (With Downsampling: 10 and for 10 iterations)

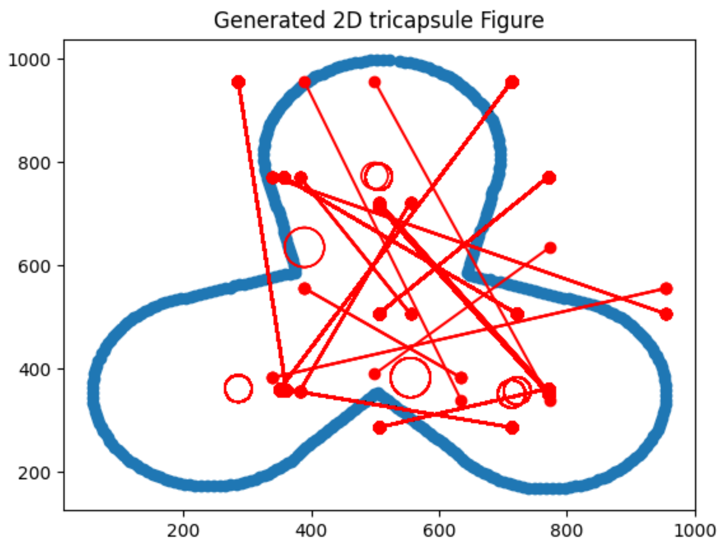

Figure 3.4(a): A Tricapsule

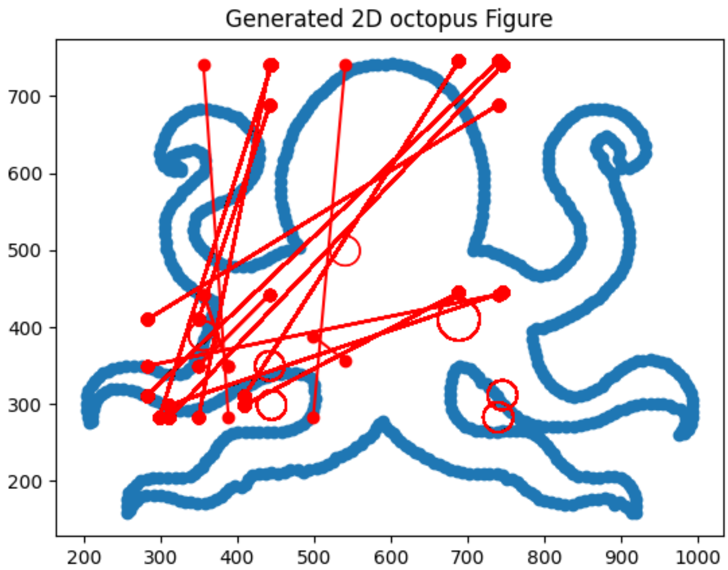

Figure 3.4(b): An Octopus

3.5 Implementation details

The following crucial input parameters are required for running the algorithm:-

1. *Downsampling rate (d):* This option takes in the no. of points per d no. of points. This is important for the stability and robustness of the algorithm – otherwise extra points may introduce excessive noise. We found a rate of 10 to satisfy our needs although this parameter can vary greatly from user to user.
2. *Neighbours (k):* The no. of neighbouring points to be considered while running a kd query. We found the optimal value to be 10.
3. *Initial radius (r):* The arbitrary initial radius which is to be given for the initial empty maximal ball to be formed. If the initial point is the first point itself, then the radius can be obtained using the compute_radius function below:-

$$\rho_p^{i+1} = \frac{d(p, \tilde{p}^i)}{2\cos\theta_p^i},$$

   We kept the default value of r to be 500 for all our experiments.

4. *Iterations (n):* An initial budget which is given to the algorithm which should be hopefully enough for the algorithm to converge to a good approximation. We found either 9 or 10 iterations to be suitable for our purposes.
5. *Workers (w):* The no. of multiprocessing workers to be used for the algorithm. Recommended value is no. of CPU cores-1.

The x, y coordinates and the corresponding normals at each point are given in .txt files which are then pre-processed to remove any duplicate or malicious points. These are then used in accordance to Algorithm 1 as proposed in section 3.2.

4 Conclusion

In this project we achieve the two major goals set out: to obtain the maximal empty ball representation for shapes (as illustrated in section 3.3) and to obtain a good approximation of input shapes using the proposed pill decomposition algorithm as shown in section 3.4.

Some particular points need to be kept in mind when designing such systems:-

- **Extent of Approximation.** This depends on the particular geometric representation being chosen and the input shape under consideration.
- **Extension to 3D based data.** This algorithm needs to significantly modified if it's to be applied to 3D data such as consideration of ray projection directions and changes in the fundamental pill structure to include voluminous data as well.
- **Real time systems.** Our codebase is already implemented in a multi-thread setting and can very easily be extended to a GPU setting as well – each of the individual KD-tree queries can be treated as parallel threads.

The man challenge has always been to find an appropriate representation, which when coupled with an efficient algorithm can be used for other real-time applications. This trade-off between better representation vs more accuracy opens up a vast arena of research especially in classic computer graphics where this subject has seen significant interest. Also, learnable representations can also be another viable option: specific representation for each type of shape may be best suited for that purpose.

5 Furthur Applications

Figure 5(a): Real time VR tracking

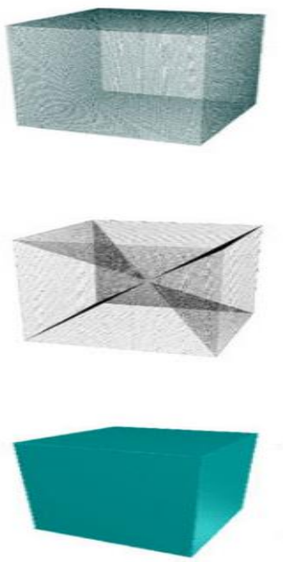

Figure 5(b): 3D Reconstruction

References

[1] Tkach A, Pauly M, Tagliasacchi A. Sphere-meshes for real-time hand modeling and tracking. ACM Transactions on Graphics (ToG). 2016 Nov 11;35(6):1-1.

[2] Katz RA, Pizer SM. Untangling the Blum medial axis transform. International Journal of Computer Vision. 2003 Nov;55(2):139-53.

[3]https://groups.csail.mit.edu/graphics/classes/6.838/F01/lectures/MedialAxisEtc/presentation/5.html

[4] Ma J, Bae SW, Choi S. 3D medial axis point approximation using nearest neighbors and the normal field. The Visual Computer. 2012 Jan;28(1):7-19.

Entire codebase is available at: https://github.com/sarosijbose/BodyTrack2D

.exe files for generating MAT from shapes:
https://github.com/jijup/Summer2022/tree/main/software