Finding Intrinsic Points on 3D Models with Using Genetic Algorithms

Altuğ Uysal
Middle East Technical University
Ankara, Turkiye

Mert Uludoğan
Middle East Technical University
Ankara, Turkiye

ABSTRACT

The identification of symmetrical points has always been a challenging task with significant applications in various fields. In this paper, we address the complexity of this issue by using an evolutionary algorithm-based approach. Our method leverages the principles of natural selection and genetics to optimize the search for symmetry points without depending on the pose of the model, reducing time complexity and improving accuracy. The proposed approach involves applying a customized fitness function to evaluate candidate points on the surface of 3D models. The fittest symmetrical points are selected for crossover and mutation, guided by the genetic algorithm. We also incorporate geodesic distance information to ensure independence from the pose of the 3D models. We evaluate the performance of our genetic algorithm-based approach using benchmark 3D models, comparing it to existing methods such as feature-based symmetry detection. The assessment includes convergence time, solution quality, and solution diversity. Our research contributes to the development of efficient and accurate techniques for identifying symmetrical points in 3D models, with potential applications in shape analysis, model reconstruction, and object recognition.

ACM Reference Format:
Altuğ Uysal and Mert Uludoğan. 2023. Finding Intrinsic Points on 3D Models with Using Genetic Algorithms. In *Proceedings of ACM Conference (Conference'17).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Symmetry is an essential characteristic in many natural and human-made objects, and its recognizing tasks are critical tasks in various applications, including computer vision, graphics, and engineering. Identifying symmetrical points in 3D models is particularly challenging because of the complexity of the models and the huge search space. Moreover, symmetry detection in 3D models has significant potential in shape analysis, model reconstruction, and object recognition. Therefore, developing efficient and accurate techniques to identify symmetrical points in 3D models is an active area of research.

Symmetry is not only valued for its aesthetic appeal but it also has practical applications in physical and algorithmic optimization techniques. For instance, The design of architectural constructions, such as LEGOs, is highly based on symmetry properties with the aim of increasing stability and efficiency [11], while in geometry processing algorithms, it can be utilized in tasks such as automatic surface mapping [3][7], surface segmentation [1], and surface embedding [14] for improved computational performance.

Symmetry can be classified into two types extrinsic symmetry and intrinsic symmetry. Extrinsic symmetry simply refers to the property of an object being symmetric about a specific axis or plane in space. It can be deductible from the pose of the model, hence its computation is less complex and easy to get more accurate results. On the other hand, intrinsic symmetry is the property of an object having symmetrical characteristics with respect to its own geometry, not depending on its orientation or position in space. Therefore, in the calculation of intrinsic symmetry points complexity has always been an issue. Intrinsic symmetry is an invaluable concept in shape analysis, model reconstruction, and object recognition besides its significant potential in various applications.

Identifying intrinsic symmetrical points in 3D models is a challenging task that requires significant computational resources and specialized techniques. The complexity of the models and the broad search space makes it difficult to identify intrinsic symmetry points using traditional methods. Moreover, the complexity of identifying symmetry points in 3D models is compounded by the absence of a standardized approach for this task.

To address this issue, we propose a genetic algorithm as a method to identify intrinsic symmetrical points in 3D models. Genetic algorithms are the outcome of the inspiration of the principles of natural selection and genetics. They are particularly useful in solving optimization problems where the search space is large and has high complexity in naive approaches. In the context of 3D modeling, genetic algorithms are proposed as a promising method to solve the identification of symmetrical points on 3D models problem with the purpose of reducing time complexity.

Previous research in this area has focused on isometric shapes, as demonstrated by the "A Genetic Isometric Shape Correspondence Algorithm with Adaptive Sampling" paper[8]. This algorithm utilized genetic algorithms and adaptive sampling techniques to find correspondences between isometric shapes, which are shapes that have the same shape but different orientations or positions in space. While this paper provides valuable insights into using genetic algorithms for symmetry point detection, there is a need to extend this research to non-isometric shapes, which are more complex and challenging to analyze. Therefore, the current study aims to build upon this previous work and improve genetic algorithms for finding symmetry points in 3D models that are not necessarily isometric.

The use of genetic algorithms for identifying symmetrical points in 3D models involves applying a fitness function to a set of candidate points which are the points extracted from the surface of 3D models. The fitness function evaluates the fitness of each candidate point based on its degree of symmetry and other relevant features. The fittest symmetrical points are then selected for crossover and mutation. However, the selection of appropriate genetic algorithm parameters and fitness functions is crucial for their success. Additionally, the time required for convergence is another challenge when utilizing genetic algorithms in 3D modeling.

This research aims to develop a customized fitness function and genetic algorithm to find intrinsic symmetrical points on 3D models efficiently and accurately. The research also aims to evaluate the performance of the genetic algorithm and compare it to other methods to enhance the accuracy and efficiency of 3D modeling.

Benchmark 3D models will be utilized to evaluate the performance of the genetic algorithm. The fitness function will be tailored to integrate symmetry principles, and the performance of the algorithm will be compared to other existing methods, such as feature-based symmetry detection methods. The study will assess the time taken for convergence, the quality of the solutions obtained, and the diversity of the generated solutions.

Utilization of distance is based on geodesic distance information of the 3D model. In this way, a point's distance to the symmetry axis will be independent of the pose of the 3D models. This geodesic information is used in the genetic algorithm's fitness function. In order to simplify calculations and have more accurate results, a single reference point on the symmetry axis of the model is used at the start.

The proposed genetic algorithm-based approach can be seen as a technique of using the power of the evolution process which is artificially configured based on the properties of the main goal. Proposing a statistically successful method can be insightful in some areas facing problems that require computations in a probabilistic space.

In summary, the identification of intrinsic symmetry points on 3D models is an essential task in various applications in computer vision, graphics, and engineering. The proposed research aims to develop an efficient and accurate method to identify symmetrical points in 3D models using genetic algorithms. The study will evaluate the performance of the genetic algorithm and compare it to other existing methods to enhance the accuracy and efficiency of 3D modeling.

2 RELATED WORK

Symmetry detection and analysis in 3D models have been the subject of extensive research due to their significance in various fields such as computer vision, graphics, medical imaging, and shape analysis. This section presents a review of relevant studies that focus on finding symmetrical points on 3D models using different algorithms and approaches.

Raviv et al. emphasize the importance of symmetry detection in practical applications, as the absence of symmetry can indicate anomalies or abnormal behavior [6]. The paper introduces a method for finding intrinsic symmetries of non-rigid shapes, addressing the limitations of traditional extrinsic symmetry descriptions. The proposed efficient method computes intrinsic symmetries of shapes, providing a basis for shape reconstruction and analysis.

Kim et al. propose an algorithm for detecting global intrinsic symmetries in 3D surface meshes. The authors make two core observations: critical points of the Average Geodesic Distance (AGD) function can robustly detect symmetry invariant point sets, and intrinsic symmetries are self-isometries of surfaces, belonging to the low-dimensional group of Möbius transformations [2]. Their algorithm generates a set of symmetric points, enumerates candidate Möbius transformations, and selects the best mappings that preserve the surface's symmetry. The stability of AGD and the low dimensionality of the Möbius group contribute to the algorithm's robustness and effectiveness.

Xu et al. recognize that while many 3D objects exhibit global symmetries, there are also prominent partial symmetries that exist only on specific parts of an object [13]. Their algorithm focuses on extracting partial intrinsic reflectional symmetries (PIRS) of 3D shapes. By developing a voting scheme and an intrinsic reflectional symmetry axis (IRSA) transform, they accentuate prominent IRSAs and extract explicit IRSA curves on the shape. The iterative refinement procedure enhances accuracy and addresses rotational symmetries, enabling the incorporation of IRSA curves into mesh segmentation schemes for more meaningful results.

Building upon their previous work, they present an algorithm for multi-scale partial intrinsic symmetry detection in 2D and 3D shapes [12]. They decouple scale extraction and symmetry extraction by employing two levels of clustering. The first level identifies significant symmetry scales using clustering of sample point pairs, and the second level extracts partial symmetries at each scale using a novel point-to-point symmetry affinity measure. The algorithm handles complex shapes with rich symmetries at multiple scales, demonstrating its effectiveness in detecting overlapping and varying symmetrical regions.

Wang and Huang propose a novel group representation of global intrinsic symmetries for 3D shapes [10]. They describe each global intrinsic symmetry as a linear transformation of functional space on shapes, leveraging the eigenfunctions of the Laplace-Beltrami operator as the basis of functional space. The group representation exhibits a block diagonal structure and can be uniquely determined from a small number of symmetric point pairs. The authors develop an efficient global intrinsic symmetry detection method capable of detecting reflectional and rotational global intrinsic symmetries with a clear group structure description.

Ovsjanikov et al. address the detection and computation of symmetries invariant up to isometry-preserving transformations [5]. Their approach transforms intrinsic symmetries into Euclidean symmetries in the signature space defined by the eigenfunctions of the Laplace-Beltrami operator. By detecting and computing isometric mappings from the shape onto itself, the algorithm proves to be computationally efficient and robust against small non-isometric deformations.

Nagar and Raman focus on the detection of intrinsic reflective symmetry in triangle meshes [4]. They extend the functional map framework to establish correspondences between functions defined on shapes, recovering point-to-point correspondences. The authors propose a closed-form solution for finding functional correspondences matrix, which significantly improves the computational efficiency of their approach. They exploit the properties of intrinsically symmetric shapes, such as the symmetry of the shortest geodesic length between symmetric points and the even/odd nature of eigenfunctions of the Laplace-Beltrami operator. Their method is invariant to the ordering of eigenfunctions and demonstrates excellent performance on datasets like SCAPE and comparable results with state-of-the-art methods on the TOSCA dataset.

These studies collectively contribute to the field of symmetry detection and analysis in 3D models, providing valuable insights and algorithms for identifying symmetrical points. However, each approach has its strengths and limitations, and the choice of method depends on the specific requirements of the application. In the following sections, we will build upon these existing works to propose our own approach for finding symmetrical points on 3D models using a genetic algorithm, aiming to overcome the limitations and explore new possibilities in this domain.

3 METHOD

Our methodology follows a pattern from processing the 3D model information to feeding the genetic algorithm with extracted data. Initially, a manual point on the 3D model as user input is used to determine a point that is not affected by reflection. This point is required for the sake of simplification in sampling the meshes equally from the model's symmetrical axis. Our method uses the same amount of points to generate the initial population from both sides. In order not to take the risk of increasing complexity, we chose to obtain this point.

Problem Formulation: We define the problem as follows: Given a 3D model represented as a surface mesh, our objective is to identify symmetrical points on the model that exhibit intrinsic symmetries. We aim to detect both reflectional and rotational symmetries, accounting for potential variations in symmetry across different scales and orientations.

Representation: To apply the genetic algorithm, we need to define the representation of an individual in the population. In our case, an individual represents a set of candidate symmetrical points on the 3D model. Each individual represent 3D dimension values: x, y, z; corresponding to a vertex in the mesh. This is our gene in our application. We have two important "genes", input vertices, corresponding to four reference points used in calculating errors of symmetry candidates. Two of these reference points are taken to represent points which are non-reflective for symmetry, so the user must use its instinctive approach to choose two points on symmetry axis. With these values, we split the mesh two regions. In algorithm side, we will say them left, for "Reference Vertex", and right, for "Candidate Vertex". The other input points are from the opposite sites of the previously chose symmetry axis. This will be used to build an error function.

Fitness Function: The fitness function does computation on distances between the vertex and the input references given for symmetry axes. This fitness function will calculate a value which represents the error of the symmetry for vertices. These error values are our descriptor to assess the symmetry feature of current pair against the nature of the Genetic Algorithm environment. Two different techniques are used: "Relative Error", "Relative Squared Error". We calculate errors for each input on symmetry axes, we have two. Two different errors from two different input points add up linearly.

Fitness Function - Relative Squared Error

```
EVALUATE_FITNESS( DIST_INPUT_LEFT, DIST_INPUT_RIGHT ):
    IF DIST_INPUT_RIGHT == 0:
        FITNESS_SCORE = infinity
    ELSE:
        FITNESS_SCORE =
            ( DIST_INPUT_LEFT - DIST_INPUT_RIGHT )
                            /
                    DIST_INPUT_RIGHT
```

Fitness Function - Relative Squared Error

```
EVALUATE_FITNESS( DIST_INPUT_LEFT, DIST_INPUT_RIGHT ):
    IF DIST_INPUT_RIGHT == 0:
        FITNESS_SCORE = infinity
    ELSE:
        FITNESS_SCORE =
            ( DIST_INPUT_LEFT - DIST_INPUT_RIGHT )^2
                            /
                    DIST_INPUT_RIGHT^2
```

Genetic Operators: We employ three main genetic operators to evolve the population: selection, crossover, and mutation.

Selection: The selection operator determines which individuals from the population will be selected as parents for reproduction. We use a fitness-based selection strategy, giving higher chances of selection to individuals with lower fitness scores. This bias towards fitter individuals promotes the preservation and propagation of favorable traits. We use error values of the pairs and filter them against such thresholds: "Success error threshold", "Mutation probability", "Mutation barrier threshold", and "Outlier error threshold". "Success error threshold" corresponds to the fitness value of the given symmetry inputs. This value is compared with the fitness values of pairs to filter them for acceptance. "Mutation probability" is the a value between 0 and 1, representing the probability of mutation. "Mutation barrier thresholds" and "Outlier error threshold" are two z values, corresponding to the Z value which stands for the upper limit of mutation barrier and lower limit for outliers, respectively. Moreover, all the errors of the population have been used to create "Z errors" to filter the pairs against thresholds.

Selection Function

```
SELECTION():
    for each PAIRS:
        IF ( Pair.ERROR < SUCC_ERROR_TH )
                    and
            Pair.POSITION is RIGHT:
          SYMMETRY_DETECTED # This pair is now immutable

        ELSE:
            IF ( RANDOM < MUTATION_PROB
                        and
                Pair.Z_ERROR > MUTATION_BARRIER_Z )
                        or
            ( Pair.Z_ERROR > OUTLIER_BARRIER_Z ):
                MUTATION_OCCUR
        ELSE:
                CROSSOVER_OCCUR
```

Crossover: The crossover function uses the clustered distances. The distances are sorted ascending ordered and split to 5 different regions. The candidate vertex have region indices for two symmetry axes input. For each input, region index of the candidate vertex is go up or down towards to region index of the reference vertex. Then the candidate vertex is replaced with the vertex in this new region. The points of this approach is to cluster the current vertices according to their distances. Thus, we can resemble the candidate vertex to reference vertex.

Crossover Function

```
CROSSOVER(pair):
    Ref_V = &pair[0]
    Cand_V = &pair[1]

    IF Ref_V.REGION_INDEX > Cand_V.REGION_INDEX:
        Cand_V.REGION_INDEX +=1
    ELSE IF Cand_V.REGION_INDEX > Ref_V.REGION_INDEX:
        Cand_V.REGION_INDEX -= 1

    New_V = RANDOM from REGION[Cand_V.REGION_INDEX]

    Cand_V = New_v
```

Mutation: The mutation operator introduces random changes into individual solutions. It helps prevent premature convergence by exploring new regions of the search space. In our approach, we apply mutation to pairs with random probability, "Mutation Probability", 0.01 and we kept it constant. The randomness nature of the genetic algorithm is highly depending on mutation so we also kept the algorithm simple.

Mutation

```
MUTATION(pair):
    Cand_V = &pair[1]
    Cand_V = RANDOM from RIGHT_VERTICES
```

Genetic Algorithm Workflow: Our method follows a standard genetic algorithm workflow:

a. Initialization: We initialize the population with a set of randomly generated individuals, each representing a potential set of symmetrical points.

b. Evaluation: We evaluate the fitness of each individual in the population using the fitness function described earlier.

c. Selection: We select individuals from the population as parents for reproduction based on their fitness scores.

d. Crossover: We perform crossover operations on the selected parents to generate offspring individuals with combined genetic information.

e. Mutation: We apply mutation to the offspring individuals to introduce random changes.

f. Replacement: We replace some individuals in the population with newly generated offspring.

g. Termination: We repeat steps b to f for a fixed number of generations or until a termination criterion is met, such as reaching a satisfactory level of symmetry or convergence.

Symmetrical Point Extraction: Once the genetic algorithm terminates, we extract the set of symmetrical points from the fittest individual in the final population. These symmetrical points represent the detected intrinsic symmetries of the 3D model.

Post-processing and Refinement: To enhance the accuracy of the detected symmetrical points, we can apply post-processing techniques such as noise filtering, outlier removal, and local refinement. These steps help improve the precision and reliability of the symmetrical point identification.

Implementation Details: We implement our method using a suitable programming language (C++) and relevant libraries for 3D mesh processing and genetic algorithms. We carefully tune the parameters of the genetic algorithm, including population size, crossover and mutation rates, and termination criteria, to achieve a balance between exploration and exploitation of the search space.

In summary, our method utilizes a genetic algorithm to discover symmetrical points on 3D models. By iteratively evolving a population of candidate solutions, guided by a fitness function, we aim to converge towards optimal sets of intrinsic symmetrical points. The algorithm employs selection, crossover, and mutation operations to explore the search space and refine the detected symmetries.

Our general algorithm flow starts with a uniform sampling of the vertices of the model and initializing population. Then will use evolutionary mechanisms to converge to a binary result of the points in the decision of their symmetrical properties.

Test Algorithm Workflow

```
    INPUT: model, a point on model
    OUTPUT: symmetry axis for the model

    Model = LoadModel(model)
    SampledPoints = Sample(model.vertices)
    Population = InitializePopulation(SampledPoints)
    GeneticAlgorithm(Model, SampledPoints)
```

Genetic Algorithm

```
Initialize candidate regions and reference regions
Initialize pairs between regions

While REACHED_SYMM_TH or MAX_GENERATION:
    Evaluate Fitness Values for Generation
    OPERATIONS = [Selection, Mutation, Crossover]
    for OP in OPERATIONS:
        OP(pair) if OP_VALID for pair
    Generation = Next_Generation
```

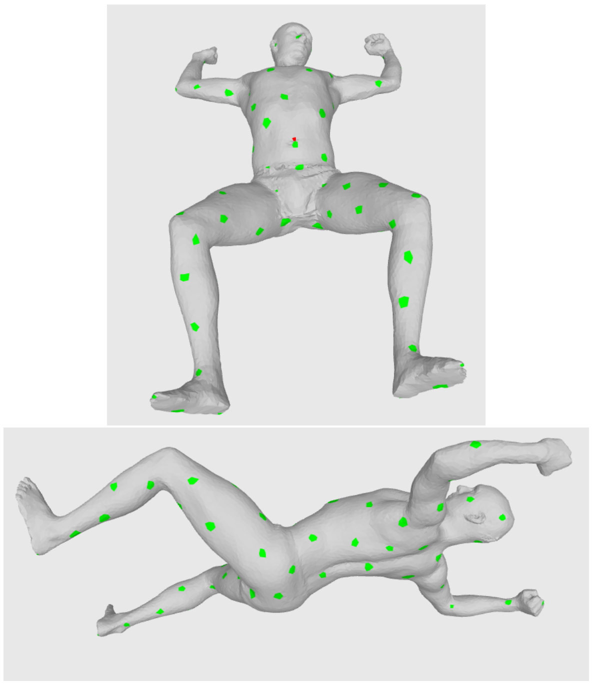
Figure 1: Evenly Spaced Samples

4 EXPERIMENT

Experiment Setup: First, we read the 3D model and preprocess it for analysis. As part of the preprocessing step, we calculate the distances between each vertices to both input points standing for symmetry axis references.

Point Sampling: To obtain a representative set of points for symmetry detection, we tried to use different methods to sample points from the 3D model. Evenly-spaced sampling and uniform sampling methods were the most successful methods to split equally. In evenly-spaced sample (Figure 1), each point is selected based on its proximity to the center of mass. We colorize the nearest meshes to the other meshes to visually represent the sampling process based on sample size.

Genetic Algorithm Integration: Next, we will attempt to feed the genetic algorithm with the sampled points for symmetry detection. However, at the current stage, we have not yet achieved conclusive results in terms of identifying the symmetrical points accurately.

Moving forward, we have outlined the following steps to improve the performance of the method:

4.1 Parameter Configuration

We have parameters in order to configure the natural environment. Some of the parameters are kept constant with purpose:

a) "Mutation probability" is 0.01. Some other values are also tested. However, randomness limiting factors in selection process are more dominant in the algorithm. Therefore, configuring other limiting factors is more reasonable. Moreover, there are more data to analyze and find benchmark values for parameters in fitness, selection and crossover processes.

b) "Mutation barrier threshold" is -1.96, which is for if the pair is considerably close to inputs, there is no need to mutate them. We can preserve some pairs with this approach, and so the pairs which have higher potential to be symmetric are not lost.

c) "Outlier error threshold" is 1.96, which stands for the acceptance of %97.5 of the errors of pairs. It rejects the very long distances due to their huge effects on statistical variables depending on errors. This approach helps for stabilizing the statistical deviations of populations over time.

4.2 Evaluation

The SCAPET[9] dataset is used for testing. We analyze the metrics of the tests. Figure 2 shows some results from different error functions and z values. The parameters are configured to show that trend line is logarithmic and the generation which the fitness value reaching significance level, in our cases it is mostly %5. The nature of the genetic algorithm is so fragile, so we have to be careful about our parameters as can be seen at the graph on the top. We used 8 sized segmented crossover function for this test. Therefore, fitness values dropped dramatically. This can be evaluated as a mix of randomness of genetic algorithm and segmentation factor, which is a limiting factor. The other graphs have more strong bonds with the randomness feature of the environment. The number of the generation for finding intrinsicly symmetric pairs is 200-300. This allows us to used a benchmark for comparing it with the naive approach. Table 1 represents average values for time durations of naive approach, and two genetic algorithms. The genetic algorithm using square error seems more effective than both naive and other genetic algorithm approach. On the other hand, genetic algorithm with linear error does not provide significant effects for time reduction.

5 CONCLUSION

In this paper, we presented a method for finding symmetrical points on 3D models using a genetic algorithm. Our approach leverages the genetic algorithm's ability to explore the search space and converge towards optimal solutions by iteratively evolving a population of candidate symmetrical point configurations. By formulating the problem of symmetry detection as an optimization task, we aimed to identify both reflectional and rotational symmetries, accounting for variations in scale and orientation.

Through a comprehensive review of relevant literature, we highlighted the importance of symmetry detection in various practical applications, such as crystallography, medical imaging, and face recognition. We discussed the challenges associated with detecting symmetries in non-rigid shapes and the significance of intrinsic symmetries in shape reconstruction and analysis.

Our method incorporates a genetic algorithm workflow, including initialization, evaluation, selection, crossover, and mutation, to iteratively refine the candidate solutions and identify symmetrical points on the 3D models. We designed a fitness function to measure the degree of symmetry exhibited by the selected points, considering factors such as alignment of normals, distance ratios, and geometric features.

Furthermore, we discussed the implementation details of our method. We also highlighted the importance of post-processing

Table 1: Algorithm Durations

Algorithm	Time(ms)
`Naive Approach`	156.37
`Genetic Algorithm - square error`	119.89
`Genetic Algorithm - linear error`	145.08

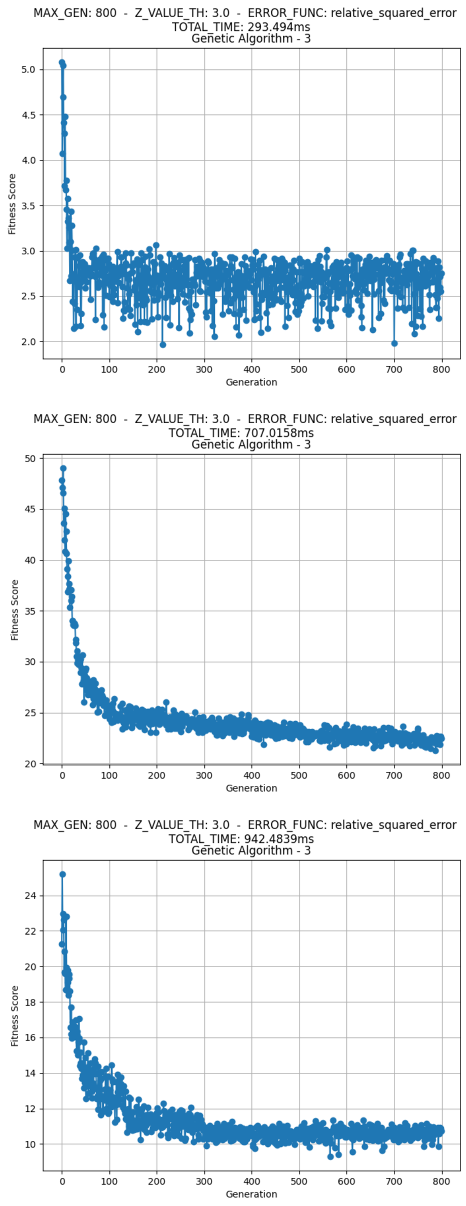

Figure 2: Generation - Fitness Value Graphs

and refinement techniques to enhance the accuracy of the detected symmetrical points.

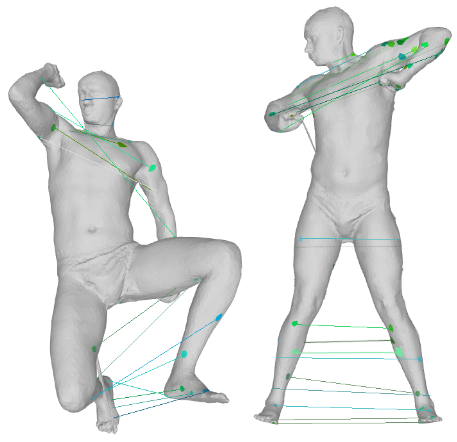

Figure 3: Symmetric Pairs

The proposed method is a promising approach for symmetrical point detection on 3D models. It provides a computational solution to the problem of identifying intrinsic symmetries, allowing for a better understanding of the underlying shape characteristics and enabling more meaningful shape analysis and reconstruction. However, it is worth noting that further research and experimentation are needed to evaluate and optimize the performance of the method on a wide range of 3D models with varying degrees of symmetry and complexity.

In conclusion, our method offers a valuable contribution to the field of shape analysis and provides a foundation for future research in symmetry detection on 3D models. By harnessing the power of genetic algorithms, we have demonstrated the potential to efficiently and effectively identify symmetrical points, facilitating advancements in diverse domains such as computer graphics, computer vision, and computational biology.

6 ACKNOWLEDGEMENTS

We would like to express our deepest gratitude to Yusuf Sahillioğlu, who has been an invaluable advisor throughout the process of conducting this research and writing this paper. His guidance, expertise, and unwavering support have played a pivotal role in shaping the direction and quality of this work.

REFERENCES

[1] Antoine Dessein, William A Smith, Richard C Wilson, and Edwin R Hancock. 2017. Symmetry-aware mesh segmentation into uniform overlapping patches. *Computer Graphics Forum* 36, 2 (2017), 95–107.
[2] Vladimir G. Kim, Yaron Lipman, Xiaobai Chen, and Thomas Funkhouser. 2010. Möbius Transformations For Global Intrinsic Symmetry Analysis. *Computer Graphics Forum* 29, 5 (2010), 1689–1700. https://doi.org/10.1111/j.1467-

8659.2010.01778.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2010.01778.x

[3] Tingbo Liu, Vladimir G Kim, and Thomas Funkhouser. 2012. Finding surface correspondences using symmetry axis curves. *Computer Graphics Forum* 31, 5 (2012), 1607–1616.

[4] Rajendra Nagar and Shanmuganathan Raman. 2018. Fast and Accurate Intrinsic Symmetry Detection. In *Proceedings of the European Conference on Computer Vision (ECCV)*.

[5] Maks Ovsjanikov, Jian Sun, and Leonidas Guibas. 2008. Global Intrinsic Symmetries of Shapes. *Computer Graphics Forum* 27, 5 (2008), 1341–1348. https://doi.org/10.1111/j.1467-8659.2008.01273.x arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2008.01273.x

[6] Dan Raviv, Alexander M. Bronstein, Michael M. Bronstein, and Ron Kimmel. 2007. Symmetries of non-rigid shapes. In *2007 IEEE 11th International Conference on Computer Vision*. 1–7. https://doi.org/10.1109/ICCV.2007.4409181

[7] Yusuf Sahillioglu. 2020. Recent advances in shape correspondence. *The Visual Computer* 36, 8 (2020), 1705–1721.

[8] Yusuf Sahillioğlu. 2018. A Genetic Isometric Shape Correspondence Algorithm with Adaptive Sampling. *ACM Transactions on Graphics* 37, 5 (2018), 175:1–175:14. https://doi.org/10.1145/3243593

[9] Yusuf Sahillioğlu and Devin Horsman. 2023. Augmented Paths and Reodesics for Topologically-Stable Matching. *ACM Transactions on Graphics* 42, 2 (2023), 1–15.

[10] Hui Wang and Hui Huang. 2017. Group Representation of Global Intrinsic Symmetries. *Computer Graphics Forum* 36, 7 (2017), 51–61. https://doi.org/10.1111/cgf.13271 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.13271

[11] Han Xu, Kwok Hong Hui, Chi-Wing Fu, and Hui Zhang. 2019. Computational Lego Technic Design. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 196.

[12] Kai Xu, Hao Zhang, Wei Jiang, Ramsay Dyer, Zhiquan Cheng, Ligang Liu, and Baoquan Chen. 2012. Multi-Scale Partial Intrinsic Symmetry Detection. *ACM Trans. Graph.* 31, 6, Article 181 (nov 2012), 11 pages. https://doi.org/10.1145/2366145.2366200

[13] Kai Xu, Hao Zhang, Andrea Tagliasacchi, Ligang Liu, Guo Li, Min Meng, and Yueshan Xiong. 2009. Partial Intrinsic Reflectional Symmetry of 3D Shapes. In *ACM SIGGRAPH Asia 2009 Papers* (Yokohama, Japan) *(SIGGRAPH Asia '09)*. Association for Computing Machinery, New York, NY, USA, Article 138, 10 pages. https://doi.org/10.1145/1661412.1618484

[14] Yoshinobu Yoshiyasu, Emi Yoshida, and Leonidas Guibas. 2016. Symmetry aware embedding for shape correspondence. *Computers & Graphics* 60 (2016), 9–22.

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009