Research Article

Artur Wróblewski and Jarosław Andrzejczak

Wave propagation time optimization for geodesic distances calculation using the Heat Method

https://doi.org/10.1515/phys-2019-0027
Received Jan 30, 2019; accepted Mar 04, 2019

Abstract: Finding the geodesic path defined as the shortest paths between two points on three-dimensional surface P is a well known problem in differential and computational geometry. Surfaces are not differentiable in a discrete way, hence known geometry algorithms can't be used directly - they have to be discretized first. Classic algorithms for geodesic distance calculation such as Mitchell-Mount-Papadimitriou method (MMP) are precise but slow. Therefore modern solutions are developed for fast calculations. One of them is Heat Method which approximates such paths with some accuracy. In this paper we propose the extension of Heat Method to reduce the approximation error. A new formula for calculating value of the parameter t (wave propagation time step) which outperforms the original one in terms of mean and median error is presented. Also, correlation between mesh properties and best wave propagation time step as well as influence of variable node spacing on heat map based method were thoroughly analysed.

Keywords: geodesic distances, Heat Method, three-dimensional meshes, MMP

PACS: 02.40.-k; 07.05.Tp; 07.05.Wr; 07.05.-t

1 Introduction

Finding the shortest paths and distances between two points on the three-dimensional surface P is a well known problem in differential and computational geometry. Such path is called geodesic path. Surfaces are not differentiable in a discrete way, hence the known differential geometry algorithms for calculating geodesic distances can not be used directly - they have to be discretized first. What is more, such surfaces can be considered as three-dimensional graphs and the algorithms from graph theory can be used. Classic algorithms for geodesic distance calculation such as Mitchell-Mount-Papadimitriou (MMP) [10] are precise but slow. Therefore modern solutions are developed for fast calculations. One of them is the Heat Method [4] which approximates such paths with some accuracy. In this paper, authors propose the extension of the Heat Method to reduce the approximation error. Also, the tests confirming the usefulness of the proposed solution are presented.

Geodesic distances play an important role in geometric analysis. There are many applications for them: three-dimensional meshes processing, registration of inhomogeneous solids, surface parameterization, segmentation or editing shape. All of them require distance calculation between any pair of points of the input mesh. Therefore, geodesic paths are used in many fields, including robotics, geographic information systems, digital circuits designing, three-dimensional meshes transformation, radiotherapy, biomedicine, dentistry and computer graphics. A wide range of applications as well as the constant demand for fast calculations of its algorithms makes this area relevant for further research.

The contributions to geodesic distances calculation research presented in this article are:

- Reduction of the mean and median approximation error in Heat Method through the use of the author's method of calculating the wave propagation time step parameter (parameter t) based on mesh properties.
- Analysis of correlation between mesh properties and best wave propagation time step as well as influence of variable node spacing on heat map based method.
- A new formula for calculating best value of parameter t (wave propagation time step) for minimal mean approximation error.
- A new formula for calculating best value of parameter t (wave propagation time step) for minimal median approximation error.

Artur Wróblewski: Institute of Information Technology, Lodz University of Technology, Łódź, Poland; Email: 214985@edu.p.lodz.pl
Jarosław Andrzejczak: Institute of Information Technology, Lodz University of Technology, Łódź, Poland; Email: jaroslaw.andrzejczak@p.lodz.pl

- Tests verifying the usefulness of proposed formulas in different scenarios as well as proposition of verification set for future research.

We start with a discussion of related work in the field of geodesic distance calculation methods in the next section. This is followed by a description of the main idea of the Heat Method and parameter t as well as the classic MMP method overview. We then present the evaluation methodology and its stages construction as well as the test sets description (basic set and verification set). Next, test results and their discussion are presented for both mean and median approximation error reduction formulas. Finally, ideas for further development and final conclusions will be given.

2 Geodesic distances

We start the analysis from the classic algorithm proposed by three authors: J. Mitchell, D. Mount and C. Papadimitriou [10] (MMP method). Popular methods based on their solution and compared to it were selected next. This way, a 13 method set was created, analysed and divided into four main categories: methods based on global wave propagation, methods based on PDE (Partial Differential Equations), graph-based methods and iterative methods (Figure 1).

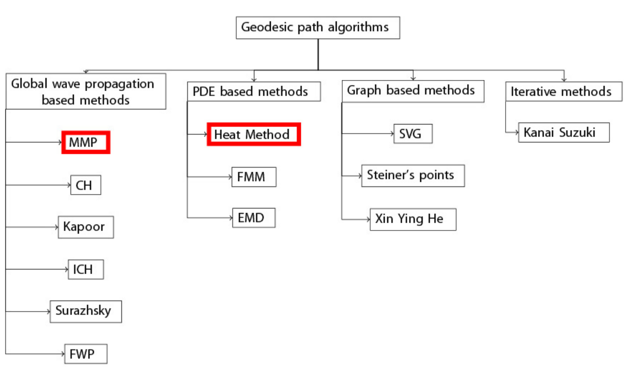


Figure 1: Geodesic distances algorithms classification. They were divided into four main categories based on used technique: methods based on global wave propagation, methods based on PDE (Partial Differential Equations), graph-based methods and iterative methods. Algorithms used in this paper are marked with red frame.

Each algorithm, as well as its family, can calculate the geodesic distance using a different approach. Due to this, solutions differ in speed, accuracy, memory requirement and time complexity. Some of the presented methods return approximate results, so it is not recommended to use them in applications demanding high accuracy. However, due to their speed they are often used in real-time systems, where accuracy is not a main priority.

The classical algorithm from global wave propagation based methods proposed by Mitchell *et al.* [10] (MMP) can calculate the shortest on-surface path in $O(n^2 logn)$ time with $O(n^2)$ space required. Surazhsky *et al.* [12] presented a detailed implementation of this method, proved that it can return results in $O(nlogn)$ in practice and extended it with merging operation to obtain a solution in a shorter time (Surazhsky method). Kappor [8] demonstrated a method which used the Dijkstra algorithm and wavefront propagation to find shortest path between pairs of points in $O(nlog^2 n)$ (Kappor method). Proposed by Chen and Han [2, 3] this method did not require Dijkstra method and can solve this problem in $O(n^2)$ time (CH method). Xin and Wang [13] greatly improved this method by filtering useless windows and maintaining priority queue which allowed this method to outperform the original version (ICH method). Recently Xu *et al.* [15] presented windows organizing method for MMP [10] and CH [2, 3] algorithms and results showed that it can improve speed by a factor of 3-10 (FWP method).

From the Partial Differential Equations (PDE) based methods group it is worth mentioning the Heat Method that was presented by Crane *et al.* [4] which used an innovative way of geodesic distances calculations using heat maps. This algorithm works on regular grids, point clouds and triangle meshes, making it a breakthrough in practice. Developed by Kimmel and Sethian [9] the Fast Marching Method (FMM) for triangulated domains can also calculate such a path but by solving Eikonal equations in $O(nlogn)$ steps. Solomon *et al.* [11] also proposed a novel method for computing shortest paths on a surface by using optimal transportation theory (EMD method). Spectral distance to geodesic distance transition was also covered.

Recently, Ying *et al.* [16] demonstrated a completely new graph based method which solves the geodesic problem from a local perspective by creating an undirected graph on which one can find the shortest path using existing efficient algorithms (SVG method). Aleksandrov *et al.* [1] proposed constructing a graph by placing m Steiner's points along each edge of the mesh therefore the shortest path can be found by using existing graph methods in $O(mnlog(mn) + nm^2)$. In contrast to well-studied single-source methods Xin *et al.* [14] presented anefficient algorithm to approximate all-pairs geodesic distances in $O(1)$ time with an additional processing step which takes $O(mn^2 logn)$ (Xin Ying He method).

Kanai and Suzuki [7] developed an iterative method which is fast, easy to implement and gives high approximation accuracy (Kanai Suzuki method). It was compared to extended CH method and it calculated distance roughly 100 times faster.

3 Heat Method and MMP

Our research was based on two existing methods for calculating distance on three-dimensional meshes:

- Heat Method [4], modern fast algorithm for geodesic distance approximation - this method was modified by the authors of this paper.
- MMP [10], classic and precise algorithm - used in evaluation as reference data for calculated distances.

Both methods are important for further analysis hence they will be described in this section in more detail.

3.1 Heat Method

The algorithm calculates distance on the surface from the source point to all other possible points. This method is simple to implement and can be used in a wide range of applications. It does not depend on mesh representation and can calculate distances on regular grids, point clouds or triangle meshes. This algorithm can be applied whenever basic derivatives known from vector calculus are available: gradient, divergence and Laplace operator. The method has three main steps:

1. Integrating the heat flow $\dot{u} = \Delta u$ for time t.
2. Evaluating the vector field $X = -\frac{\nabla u_t}{|\nabla u_t|}$.
3. Solving the Poisson equation $\Delta\phi = \nabla \cdot X$.

Every step is dependent on few differential operators and several input parameters that can be changed by the user. Available options are:

- input mesh (.obj format),
- text file with source vertices indices (single source points were used in this paper),
- output mesh (extra option to visualize isolines),
- text file with reference distance values,
- boundary condition [0 – 1], where 0 yields pure-Neumann conditions and 1 yields pure-Dirichlet conditions (according to authors on surfaces without boundaries selecting 0 is optimal),
- parameter t - time step (always positive)

Equation 1 shows value of t proposed by authors, which was based on mean spacing between vertices (nodes).

$$t = (\overline{e})^2 = \left(\frac{\sum_{i=0}^{N-1} e_i}{N}\right)^2, \text{ where} \tag{1}$$

e - edge length

N - number of edges

Impact of the parameter t on the distance calculations accuracy was pointed out as one of the further exploration possibilities. Heat method relies on time step t and such analysis could provide a different value which leads to better approximations. Moreover, because of its short calculation time, Heat Method can be used in a real-time system eg. rendering or pathfinding in computer games as well as modern solution for photorealistic computer graphics such as streamed photon mapping [17] or physically based area lightning [18]. This gives a wide range of applications of this method.

3.2 MMP

MMP method algorithm calculates the shortest path between the source point and the target point on the polygonal surface. The distance is measured using Euclidean metric and the path is on the surface. Time complexity of the algorithm is $O(n^2 logn)$ while requiring $O(n^2)$ space, where n is number of surface edges. After initial mesh processing, the distance from source point to any other point can be calculated using standard techniques in $O(nlogn)$. The actual shortest path can be returned in $O(k+logn)$, where k is the number of polygons passed-through by the path. This method also generalizes the case of many source points in order to build a Voronoi diagram on a three-dimensional surface [10].

The algorithm parameters are:

- input mesh,
- source vertex index [0 – n), where n is number of mesh vertices,
- destination vertex index (when it is not defined, algorithm calculates distances to every possible point)

MMP method was used in our research to calculate exact distances from the source point to all other possible points. Results created in this way became reference data and were used to calculate distance errors values in modified Heat Method.

4 Methodology

The aim of the performed research was to analyse impact of parameter t on the accuracy of calculated distance on surfaces with variable node spacing. Such analysis will allow to propose a new value of t which could lead to less error-prone results.

Two tests were conducted. The goal of each of them was finding the formula for calculating the value of time step t that minimized the geodesic distance approximation error in Heat Method:

- the best value of parameter t for mean approximation error
- the best value of parameter t for median approximation error

Approximation error was calculated as a difference between precise distance values calculated using MMP method and values calculated using the Heat Method with particular parameter t value. Mean approximation error represents the mean value of approximation error for all the vertices in particular mesh[1]. By analogy, median approximation error for specific mesh is calculated[2].

To achieve the goal we had to develop a process which divided tasks into smaller parts. Consequently each test had two sessions. In the first one we wanted to obtain a new formula for the parameter t (best or maximum one). Second session focused on new formula verification. All steps were as follows:

- First session:
  1. Find the lowest error (mean or median) and corresponding value of t for each mesh from basic set.
  2. Create correlation table of t value with mesh properties.
  3. Choose the most significant correlation.
  4. Develop a formula for calculating the parameter t value based on mesh properties values and their correlation with the parameter t values.
- Second session:
  1. Verify the newly created formula on basic set and verification set.
  2. Analyse the results.

For each session we prepared a set of three-dimensional meshes: basic set and verification set. Models in both sets will be triangle meshes. Basic set is a special collection with 5 meshes of one 3D model. In order to reduce the number of external factors that might affect results, we focused only on edge length diversity. Verification set is a collection of models with various shapes and mesh topology so it was possible to check the solution in many different cases.

It is worth mentioning that during our research two additional tests were conducted to find a maximum parameter t value for both: mean and median error. This maximum value was defined as the t value that produces triple value of minimum error (mean or median respectively) on at least half of all mesh vertices. It can be taken as the far right value of the search set. By using it we can be 95% sure that best parameter t value is lower than it. At the same time this value is not too big making the created set not too big. In our research it was calculated to verify if the parameter t value calculated using newly created formula is not only close to best possible value but also not bigger than the maximum value. What is more, such maximum t value can be used as a stop condition for iterative methods that are searching for parameter t value that gives minimum approximation error. Those two additional tests are not described in detail in this paper to focus on the new formula for the best parameter t value.

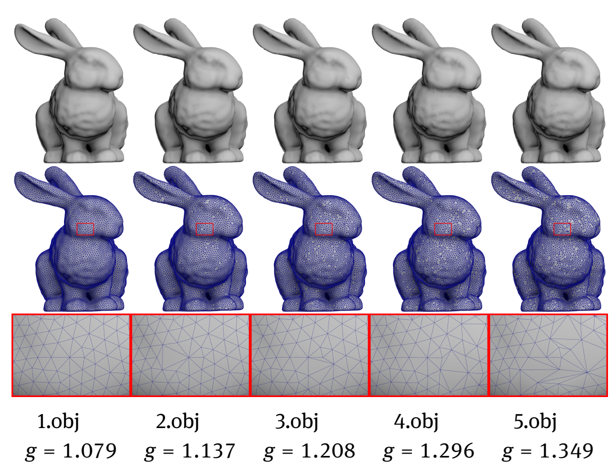


Figure 2: All meshes from basic set. Despite very similar shape, each mesh vary in edge length what is visible in anisotropy level and mesh topology. From left: original mesh 1.obj, mesh 2.obj with 3% of removed vertices, mesh 3.obj with 6% of removed vertices, mesh 4.obj with 9% of removed vertices and mesh 5.obj with 12% of removed vertices, g - anisotropy level

1 Mean approximation error will be called *mean error* later in the article for short

2 Median approximation error will be called *median error* later in the article for short

4.1 Basic set

To properly test the impact of edge length diversity on parameter t the basic set was created. The model that was selected is a well known in computer graphics rabbit - Stanford Bunny. It has 14290 vertices, 42864 edges and 28576 faces.

Meshes from this set have various anisotropy level[3] and different edge lengths. Sizes of all tested rabbits were the same - they can be inscribed in sphere with constant diameter. It was assumed that the shape of all meshes should be as close as possible to each other (Figure 2).

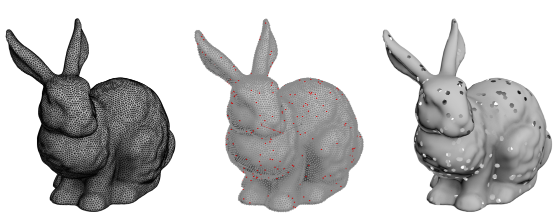
Process of creating each mesh was follows:

- n * 3% of vertices were deleted from each mesh (where n is mesh number) (Figure 3)
- Holes were filled with new polygons.
- Every newly created polygon was triangulated to keep shape of model similar to the original one.

Sometimes a set of triangles with high anisotropy level were created after triangulation of new polygons. Those cases were properly corrected to not affect shape, shading and attributes of mesh (Figure 4).

The step of size of 3% vertices from original mesh allowed to remove 12% of vertices from the last (fifth) mesh. A bigger step caused a significant deformation of shapes and according to assumptions all meshes should be as similar as they could be. What is more, from this we managed to achieve a linear increase of mesh anisotropy and standard deviation of the edge length (Figure 5).

In order to minimize additional factors that might affect results it was decided that the source point of each mesh from the basic set will be always the most protruding vertex on left side of Y axis (Figure 6).



(1) input mesh (100% vertices) **(2)** random select 3% vertices **(3)** remove selected vertices

Figure 3: Process of removing mesh vertices in basic set. From left: input mesh, randomly selected 3% of vertices, mesh with removed selected vertices

3 Anisotropy level was calculated based on formula given in [15] where detailed description of anisotropy for 3D meshes can be found.

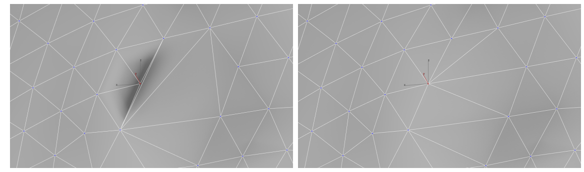


Figure 4: A vertex that negatively affects shading is visible on the left. Every vertex like that was corrected to avoid it

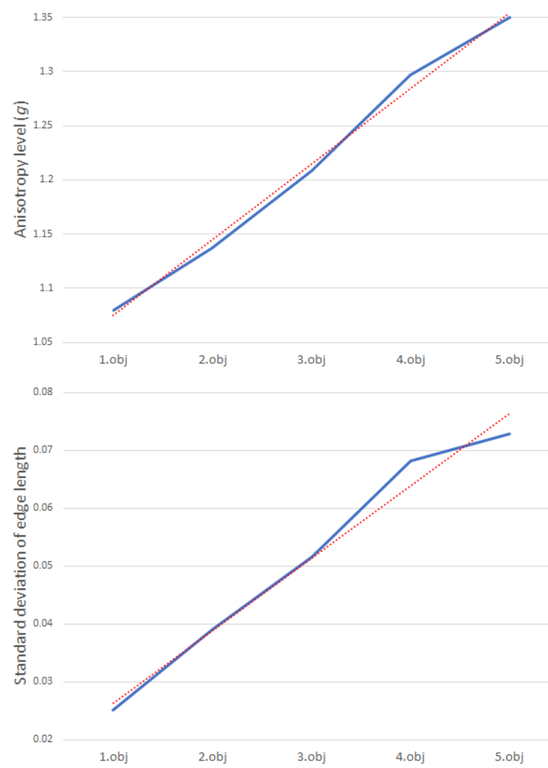


Figure 5: Linear increase of mesh anisotropy (on left) and standard deviation of the edge length (on right). Trend line is marked as Red dashed line

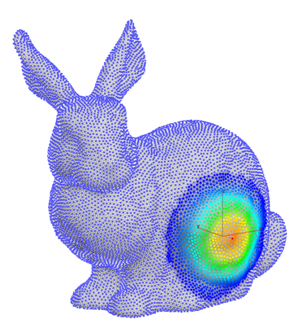


Figure 6: The most protruding vertex in Y axis was selected as source point for basic set

Table 1: Basic set meshes properties - vertices, faces and anisotropy. By removing more and more vertices from meshes they have less edges and triangles - due to that the anisotropy level increases. N_v - number of vertices; N_e - number of edges; N_f - number of faces; v_s - source vertex index; g - anisotropy level

Model name	N_v	N_e	N_f	v_s	g
1.obj	14290	42864	28576	7649	**1.079**
2.obj	13848	41538	27692	7396	**1.137**
3.obj	13398	40188	26792	7135	**1.208**
4.obj	12863	38582	25720	6816	**1.296**
5.obj	12478	37428	24952	6560	**1.349**

Table 2: Basic set meshes properties - edges. Every property value increases but it is not always linear increase - there is a significant difference between 3.obj and 4.obj, especially in variance, standard deviation and median absolute deviation. $\overline{e}$ - average edge length; $Var(e)$ - variance of edge length; σ_e - standard deviation of the edge length; D_e - median absolute deviation of the edge length

Model name	$\overline{e}$	$Var(e)$	σ_e	D_e
1.obj	0.21585	0.00062	0.02505	0.01959
2.obj	0.22093	0.00152	0.03904	0.02434
3.obj	0.22666	**0.00265**	**0.05155**	**0.03062**
4.obj	0.23393	**0.00464**	**0.06813**	**0.03933**
5.obj	0.23892	0.00531	0.07287	0.04502

Table 3: Edges length and angles for basic set meshes. The shortest edge length of each mesh remains the same. 4.obj differs from the other meshes - it has the longest edge and the biggest interior angle of triangle from entire set therefore it does not maintain linearity. min_e - the shortest edge length; max_e - the longest edge length; $\frac{min_e}{max_e}$ - ratio of shortest to longest edge; min_α - the smallest interior angle of triangle; max_α - the biggest interior angle of triangle

Model name	min_e	max_e	$\frac{min_e}{max_e}$	min_α	max_α
1.obj	0.1262	0.3296	0.38306	35.39	102.31
2.obj	0.1262	0.8303	0.15206	1.33	177.27
3.obj	0.1262	0.8848	0.14269	0.99	177.40
4.obj	0.1262	**1.2462**	0.10131	0.33	**178.96**
5.obj	0.1262	1.1172	0.11301	0.72	178.47

Despite increasing edge length diversity, the number of mesh triangles was getting smaller. We did not take processing time into account, because it should be compared only with the same amount of input data.

Table 4: Spread of edge length values for basic set. $Q_1(e)$ - first (lower) quartile of edge length; $Q_2(e)$ - second quartile (median) of edge length; $Q_3(e)$ - third (upper) quartile of edge length; QD_e - quartile deviation (semi-interquartile range) of edge length

Model name	$Q_1(e)$	$Q_2(e)$	$Q_3(e)$	QD_e
1.obj	0.20017	0.21542	0.23136	0.0155
2.obj	0.20098	0.21629	0.23331	0.0161
3.obj	0.20145	0.21709	0.23566	0.0171
4.obj	0.20216	0.21824	0.23863	0.0182
5.obj	0.20289	0.21926	0.24214	0.0196

All the data from the basic set can be found in tables Table 1, Table 2, Table 3 and Table 4. We selected those properties to test the problem from original method - impact of edge length diversity on parameter t - in the best possible. Properties considered in this paper were:

- N_v - number of vertices,
- N_e - number of edges,
- N_f - number of faces,
- v_s - source vertex index,
- g - anisotropy level,
- $\overline{e}$ - average edge length,
- $Var(e)$ - variance of edge length,
- σ_e - standard deviation of the edge length,
- D_e - median absolute deviation of the edge length,
- min_e - the shortest edge length,
- max_e - the longest edge length,
- $\frac{min_e}{max_e}$ - ratio of shortest to longest edge,
- min_α - the smallest interior angle of triangle,
- max_α - the biggest interior angle of triangle,
- $Q_1(e)$ - first (lower) quartile of edge length,
- $Q_2(e)$ - second quartile (median) of edge length,
- $Q_3(e)$ - third (upper) quartile of edge length,
- QD_e - quartile deviation (semi-interquartile range) of edge length

4.2 Verification set

In order to check the proposed solution in different conditions a verification set consisting 30 models with various shapes were created. Each of the chosen mesh was often used in many papers and analysis of computer graphics algorithms[4]. Mostly they were created by scanning real objects with three-dimensional scanners. Next, from point

4 Verification set can be accessed at DOI: 10.13140/RG.2.2.34777.77923

Table 5: Selected properties of verification set. g - anisotropy level; $\bar{e}$ - average edge length; $Var(e)$ - variance of edge length; σ_e - standard deviation of the edge length; max_e - the longest edge length

Model name	g	$\bar{e}$	$Var(e)$	σ_e	max_e
(1) armadillo	1.7725	0.503	0.0421	0.205	2.5598
(2) bee	1.5461	0.461	0.0419	0.204	2.6388
(3) bimba	1.6924	0.552	0.0422	0.205	1.4478
(4) buddha	2.5003	0.682	0.1419	0.376	3.6185
(5) camel	1.9755	0.213	0.0221	0.148	1.0632
(6) cane	1.2627	0.125	0.0015	0.039	0.6723
(7) cow	2.5022	0.437	0.1266	0.355	3.1916
(8) david	3.0191	0.340	0.0620	0.249	7.2493
(9) doll	1.5435	0.216	0.0188	0.137	2.4440
(10) duck	1.6267	0.184	0.0048	0.069	1.0212
(11) fandisk	1.3409	0.100	0.0005	0.023	0.2487
(12) fandisk2	1.3834	1.033	0.0796	0.282	2.5778
(13) fertility	1.0704	0.107	0.0003	0.018	0.1622
(14) frog	1.9705	0.298	0.1038	0.322	5.9300
(15) gargoyle	1.3510	0.164	0.0032	0.057	1.0673
(16) genus3	1.6391	0.773	0.0487	0.22	1.6547
(17) homer	1.8007	0.344	0.0475	0.218	2.5818
(18) horse	1.4912	0.096	0.0007	0.028	0.7754
(19) kitten	1.0810	0.218	0.0006	0.025	0.3157
(20) lion	3.3114	0.187	0.0453	0.212	5.4070
(21) lucy	2.1592	0.129	0.0036	0.06	0.8617
(22) max planck	1.7776	0.624	0.0559	0.236	2.9817
(23) monk	3.266	0.153	0.0146	0.121	2.1356
(24) mouse	1.6792	0.230	0.0087	0.093	0.9612
(25) pegasus	1.9811	0.561	0.0696	0.263	2.9984
(26) rockerarm	1.2696	0.123	0.0039	0.062	0.8560
(27) suzanne	1.6780	1.029	0.5095	0.713	4.4345
(28) teapot	2.6823	0.483	0.0809	0.284	0.1318
(29) torso	1.9750	0.204	0.1114	0.105	0.9234
(30) vaselion	1.6263	0.061	0.0034	0.058	5.4038

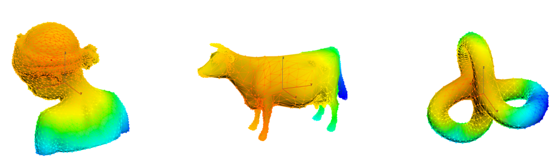


Figure 7: Examples of source points in verification set. The farther from the source, the cooler the color gets

clouds they were converted to triangle meshes. All of them (as those from the basic set) were the same size - they can be inscribed in sphere with constant diameter (Figure 8).

The number of faces of each model significantly differed and ranged from 984 to 400000. Due to this each mesh varied by eg. number of vertices, edges, anisotropy level, the smallest and biggest interior angles of triangles. Some properties of those meshes are presented in table Table 5.

In the verification set, the first vertex from the mesh vertices array was selected as the source point. Thanks to that we introduced random selection of propagation point (Figure 7).

5 Results and analysis

The results and their analysis will be divided into two parts. The aim of each part was analogical to the aim of the tests

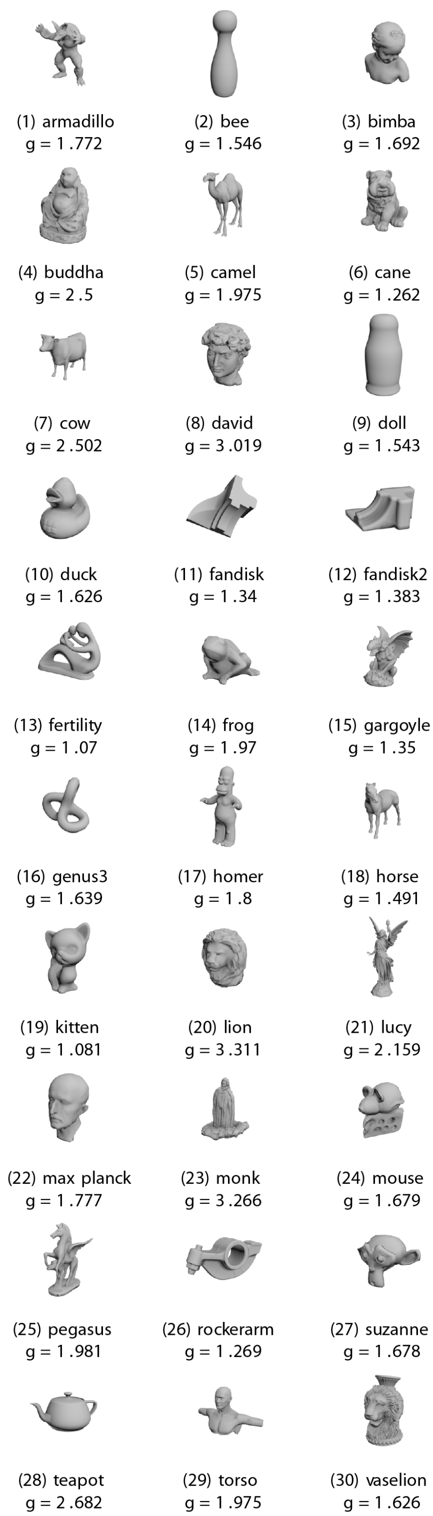


Figure 8: Verification set consisted of 30 models with various shapes and mesh topologies

described in previous section - which was finding the formula for calculating:

- the best value of parameter t for mean approximation error
- the best value of parameter t for median approximation error

5.1 Best value of t for mean error

5.1.1 Minimum error

First, the approximation error values for the following parameter t values were calculated. From them, the mean error was provided for each mesh from basic set. The minimum mean error for a given mesh is the global minimum of the function of this dependence (mean error and t value, Figure 9). Parameter t takes only positive values. The graph of such a function is similar to modulus of logarithmic function. Thanks to that we can be sure that the minimum occurs only once and if the value of t will be greater than it, average error will increase.

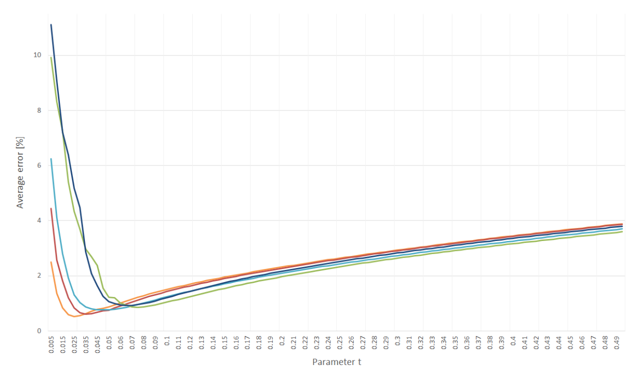


Figure 9: Parameter $t \mapsto$ mean error graph for each mesh from basic set. It is similar to modulus of logarithmic function - there is only one global minimum

We have noticed that in the basic set, extremum of a function occurs at small values of parameter t. It is worth mentioning that the minimum mean error of each mesh is getting larger, but the value of t does not converge with it (Table 6).

5.1.2 Correlation table - mesh property choosing

As a next step, correlation coefficients between mesh properties and parameter t that is giving the minimum mean error were calculated. For that purpose the Pearson correlation was used. The shortest edge length was not used

Table 6: Minimum mean errors and corresponding parameter t values for each mesh from basic set. Parameter t does not increase linearly with minimum mean error value. t - time step; $min_{\overline{\eta}}$ - minimum mean error; $t_{min_{\overline{\eta}}}$ - parameter t giving the minimum mean error

Model name	$min_{\overline{\eta}}$ [%]	$t_{min_{\overline{\eta}}}$
1.obj	0.53031	0.0254
2.obj	0.60318	0.0361
3.obj	0.7676	0.0486
4.obj	0.85488	0.0749
5.obj	0.92221	0.064

in correlation table because it is the only property that remains the same for each mesh. The strongest positive correlation was with standard deviation of the edge length - almost 96%. This property was used as the most significant in the developed formula (Table 7).

Table 7: Pearson correlation coefficients between mesh properties and parameter t that is giving the minimum mean error. $\rho(t_{min_{\overline{\eta}}})$ - correlation with parameter t that is giving the minimum mean error; $t_{min_{\overline{\eta}}}$ - parameter t that is giving the minimum mean error; σ_e - standard deviation of the edge length; $Var(e)$ - variance of edge length; max_e - the longest edge length; g - anisotropy level; $\overline{e}$ - average edge length; D_e - median absolute deviation of the edge length; $Q_2(e)$ - second quartile (median) of edge length; $Q_1(e)$ - first (lower) quartile of edge length; $Q_3(e)$ - third (upper) quartile of edge length; QD_e - quartile deviation (semi-interquartile range) of edge length; max_α - the biggest interior angle of triangle; min_α - the smallest interior angle of triangle; $\frac{min_e}{max_e}$ - ratio of shortest to longest edge; v_s - source vertex index; N_v - number of vertices; N_e - number of edges; N_f - number of faces

Mesh property	$\rho(t_{min_{\overline{\eta}}})$
$t_{min_{\overline{\eta}}}$	1
σ_e	0.9593
$Var(e)$	0.9501
max_e	0.942
g	0.9346
$\overline{e}$	0.9277
D_e	0.9254
$Q_2(e)$	0.9076
$Q_1(e)$	0.8999
$Q_3(e)$	0.8827
QD_e	0.8757
max_α	0.6933
min_α	−0.6956
$\frac{min_e}{max_e}$	-0.7973
v_s	−0.9201
N_v	−0.9279
N_e	−0.928
N_f	−0.9281

5.1.3 Developing formula and verification

Next, a formula to calculate the value of t dependent on standard deviation of the edge length was developed (Eq. 2).

$$t = \sigma_e = \sqrt{\frac{\sum_{i=0}^{N-1}(e_i - \overline{e})^2}{N}}, \text{ where:} \tag{2}$$

e - edge length

N - number of edges

$\overline{e}$ - average edge length

The newly developed formula was tested on the basic and verification sets. Figure 10 shows percentage gain using the proposed value of t based on standard deviation of the edge length. Gain was calculated by dividing two mean approximation error values: one obtained using parameter t value calculated with the new formula (Eq. 2) and the second one obtained using t value from original formula (Eq. 1). We choose a 95% confidence interval. For 21 models we can be 95% certain that we got a lower mean error then the original formula used in [4]. Original value of t provided lower mean error for 10 models. Only for 4 cases we cannot be sure which solution is better.

5.1.4 Analysis

Figure 11 shows how best, original and calculated using the new formula parameters t are shaped. For 16 models the value of best parameter t was between the original and proposed one, for 10 models it was smaller than both of them and for 9 it was larger. Extra attempt was made to adapt formula for these cases. We developed a second formula based on the average of standard deviation and average edge length (Eq. 3).

$$t = \frac{(\overline{e})^2 + \sigma_e}{2} = \frac{\left(\frac{\sum_{i=0}^{N-1} e_i}{N}\right)^2 + \sqrt{\frac{\sum_{i=0}^{N-1}(e_i - \overline{e})^2}{N}}}{2}, \tag{3}$$

where:

e - edge length

N - number of edges

$\overline{e}$ - average edge length

σ_e - standard deviation of the edge length

By using the improved formula (Eq. 3) we managed to get better results than with the first one (Eq. 2). In fact, we still can be 95% certain that we got lower mean errors for 21 models (in most cases gain value is higher than 20%), but this time original value of *t* is better in only 4 cases. For 10 models we cannot be sure which solution is better (Figure 12).

5.2 Best value of *t* for median error

5.2.1 Minimum error

For each mesh from basic set the minimum median error and corresponding value of *t* were found. Median errors

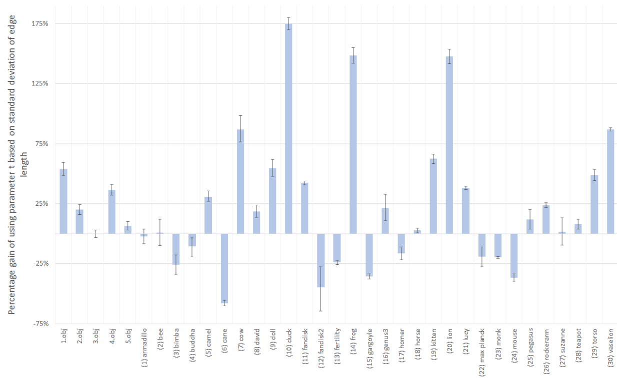


Figure 10: Percentage gain within the meaning of mean approximation error of using parameter *t* calculated based on standard deviation of the edge length. For 10 models it allows to obtain lower mean errors. Gain was calculated by dividing two mean approximation error values: one obtained using parameter *t* value calculated with the new formula (Eq. 2) and the second one obtained using *t* value from original formula (Eq. 1). 95% confidence interval values are also presented

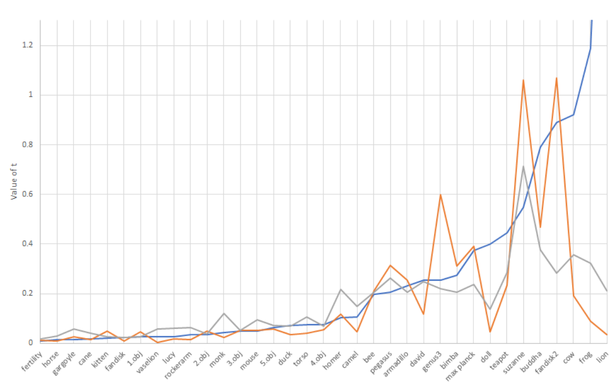


Figure 11: Best, original and proposed parameters *t* for each mesh from basic and verification sets. For many models value of best parameter *t* is between original and proposed value.

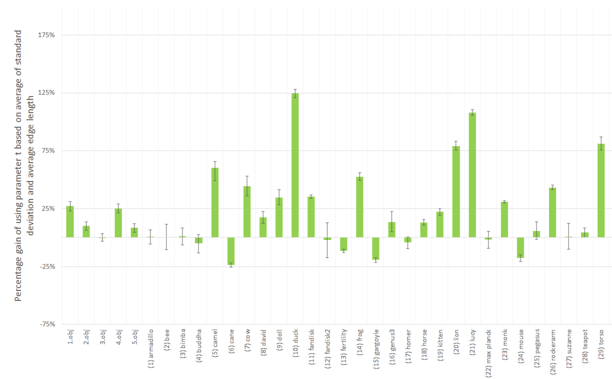


Figure 12: Percentage gain within the meaning of mean approximation error of using parameter *t* calculated based on average of standard deviation and average edge length. Only for 4 models original value of *t* allows to obtain lower mean errors. Gain was calculated by dividing two mean approximation error values: one obtained using parameter *t* value calculated with the new formula (Eq. 3) and the second one obtained using *t* value from original formula (Eq. 1). 95% confidence interval values are also presented

are smaller than mean errors but similarly parameters *t* do not converge with them. It is worth to notice that minimum median error for 4.obj is smaller than for 3.obj (Table 8).

Table 8: Minimum median errors and corresponding parameter *t* values for each mesh from basic set. $min_{\tilde{\eta}}$ - minimum median error; $t_{min_{\tilde{\eta}}}$ - parameter *t* giving the minimum median error

Model name	$min_{\tilde{\eta}}$ [%]	$t_{min_{\tilde{\eta}}}$
1.obj	0.37351	0.0246
2.obj	0.45332	0.0365
3.obj	**0.60543**	0.0553
4.obj	**0.59709**	0.0815
5.obj	0.70536	0.0613

5.2.2 Correlation table - mesh property choosing

Eight of the mesh properties strongly correlate with parameter *t* that is giving the minimum median error (over 80% - positive correlation). It is worth mentioning that the order of properties is almost the same as for mean error - only max_e switched position with $Var(e)$. Standard deviation of the edge length was selected as most significant one - 96% positive correlation (Table 9).

Table 9: Pearson correlation coefficients between mesh properties and parameter t that is giving the minimum median error. $\rho(t_{min_{\tilde{\eta}}})$ - correlation with parameter t that is giving the minimum median error; $min_{\tilde{\eta}}$ - parameter t that is giving the minimum median error; σ_e - standard deviation of the edge length; max_e - the longest edge length; $Var(e)$ - variance of edge length; g - anisotropy level; $\overline{e}$ - average edge length; D_e - median absolute deviation of the edge length; $Q_2(e)$ - second quartile (median) of edge length; $Q_1(e)$ - first (lower) quartile of edge length; $Q_3(e)$ - third (upper) quartile of edge length; QD_e - quartile deviation (semi-interquartile range) of edge length; max_α - the biggest interior angle of triangle; min_α - the smallest interior angle of triangle; $\frac{min_e}{max_e}$ - ratio of shortest to longest edge; v_s - source vertex index; N_v - number of vertices; N_e - number of edges; N_f - number of faces

Mesh property	$\rho(t_{min_{\tilde{\eta}}})$
$t_{min_{\tilde{\eta}}}$	1
σ_e	0.9113
max_e	**0.8956**
$Var(e)$	**0.8862**
g	0.8708
$\overline{e}$	0.86
D_e	0.8548
$Q_2(e)$	0.832
$Q_1(e)$	0.826
$Q_3(e)$	0.799
QD_e	0.7897
max_α	0.7017
min_α	−0.705
$\frac{min_e}{max_e}$	−0.7967
v_s	−0.8518
N_v	−0.8638
N_e	−0.8639
N_f	−0.8641

5.2.3 Formula developing and verification

With usage of linear regression we developed a formula to calculate the value of t dependent on standard deviation of the edge length (Eq. 4). The proposed equation could be simplified (by omitting the constant) but it stayed unchanged during tests.

$$t = 1.01204\sigma_e - 0.0001 \tag{4}$$
$$= 1.01204 * \sqrt{\frac{\sum_{i=0}^{N-1}(e_i - \overline{e})^2}{N}} - 0.0001, \text{ where:}$$

e - edge length

N - number of edges

$\overline{e}$ - average edge length

The newly developed formula was tested on the basic and verification sets. Figure 13 shows the percentage gain using proposed value of t based on standard deviation of the edge length. Gain was calculated by dividing two median approximation error values: one obtained using parameter t value calculated with the new formula (Eq. 4) and the second one obtained using t value from original formula (Eq. 1). We choose 95% confidence interval. For 19 models we can be 95% certain that we got lower a median error then the original formula used in [4]. The original value of t provided lower median errors for 11 models. Only for 5 cases we cannot be sure which solution is better.

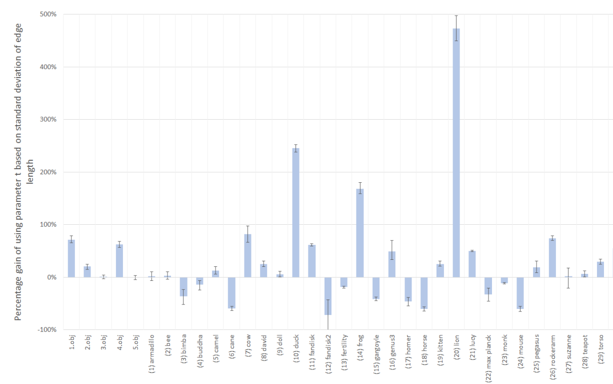


Figure 13: Percentage gain within the meaning of median approximation error from using parameter t based on standard deviation of the edge length. For 11 models it allows to obtain lower median errors. Gain was calculated by dividing two mean approximation error values: one obtained using parameter t value calculated with the new formula (Eq. 4) and the second one obtained using t value from original formula (Eq. 1). 95% confidence interval values are also presented

5.2.4 Analysis

Graphs of best, original and proposed parameters t were similar for median error (Figure 14) and mean error (Figure 11). We decided to check if the solution applied for finding formula for parameter t that is giving minimum mean error could be used here. For 13 models the value of best parameter t was between the original and proposed one, for 12 models it was smaller than both and for 10 it was larger. Theoretically adjusting the formula should be profitable (Eq. 5).

$$t = \frac{(\overline{e})^2 + 1.01204\sigma_e - 0.0001}{2} \tag{5}$$

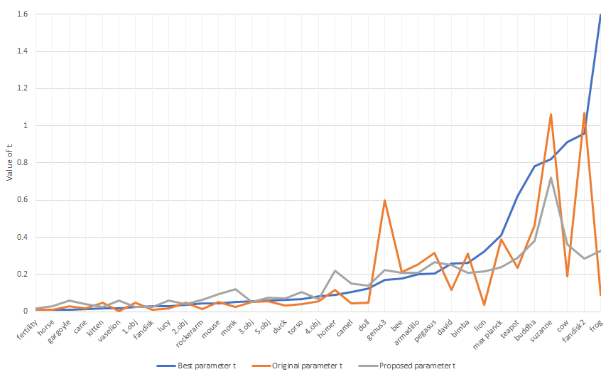


Figure 14: Best, original and proposed parameters t for each mesh from basic and verification sets

$$= \frac{\left(\frac{\sum_{i=0}^{N-1} e_i}{N}\right)^2 + 1.01204 * \sqrt{\frac{\sum_{i=0}^{N-1} (e_i - \overline{e})^2}{N}} - 0.0001}{2},$$

where:

e - edge length

N - number of edges

$\overline{e}$ - average edge length

By using the improved version of the formula (Eq. 5) we managed to get better results than with the previous one (Eq. 4). This time we can be 95% certain that we got lower median errors for 18 models (one less than results with usage of Eq. 4, in most cases gain value is higher than 30%). This time the original value of t is better in only 6 cases. For 11 models we cannot be sure which solution is better (Figure 15).

6 Conclusion

The aim of our research was to analyse and modify a method which used heat maps for calculating distances on three-dimensional surfaces. Heat Method algorithm was modified by using a new formula for calculating the value of parameter t. During this work, over a dozen mesh properties related to variable node spacing were analysed in correlation with parameter t. In result, two new formulas for calculating parameter t based on the average of standard deviation and average edge length for a particular mesh were proposed. Then, tests were conducted to verify the proposed solution with various meshes. Results have shown that both new formulas proposed in this article provide better or similar results for mean and median approximation errors than original one in most cases (gain value usually bigger than 20-30%). Thanks to that, we can use a fast algorithm for calculating geodesic distance with higher accuracy than before. Also, the verification set proposed in this paper can be used for future research in the area of geodesic distance calculation.

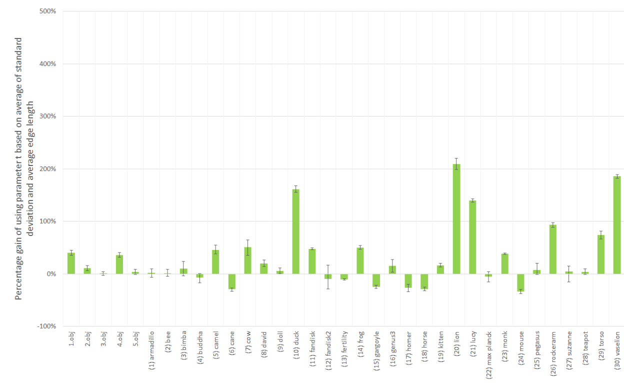


Figure 15: Percentage gain within the meaning of median approximation error of using parameter t based on average of standard deviation and average edge length. Only for 6 models original value of t allows to obtain lower median errors. Gain was calculated by dividing two mean approximation error values: one obtained using parameter t value calculated with the new formula (Eq. 5) and the second one obtained using t value from original formula (Eq. 1). 95% confidence interval values are also presented.

References

[1] Aleksandrov L., Lanthier M., Maheshwari A., Sack J-R., An ϵ-approximation for weighted shortest paths on polyhedral surfaces, Proceedings of the 6th scandinavian workshop on algorithm theory, 1998

[2] Chen J., Han Y., Shortest paths on a polyhedron, Proceedings of the sixth annual symposium on computational geometry. ACM, 1990

[3] Chen J., Yijie H., Shortest paths on a polyhedron, Part I: Computing shortest paths International Journal of Computational Geometry and Applications, 1996

[4] Crane K., Weischedel C., Wardetzky M., Geodesics in heat: A new approach to computing distance based on heat flow, ACM Trans Graph, 2013

[5] Dijkstra E., A note on two problems in connexion with graphs, Numerische Mathematik, 1959

[6] Geodesic on meshes, https://houdinigubbins.wordpress.com/2017/05/02/geodesics-on-meshes/

[7] Kanai T., Suzuki H., Approximate shortest path on a polyhedral surface based on selective refinement of the discrete graph and its applications, Geometric modeling and processing, 2000

[8] Kapoor S., Efficient computation of geodesic shortest paths, Proceedings of the Thirty-First Annual ACM Symposium on Theory of Computing, 1999

[9] Kimmel R., Sethian JA., Computing geodesic paths on manifolds, Proc Natl Acad Sci USA, 1998

[10] Mitchell JSB., Mount DM., Papadimitriou CH., The discrete geodesic problem, SIAM J Comput, 1987
[11] Solomon J., Rustamov R., Guibas L., Butscher A., Earth Mover's distances on discrete surfaces, ACM Trans Graph 2014
[12] Surazhsky V., Surazhsky T., Kirsanov D., Gortler SJ., Hoppe H., Fast exact and approximate geodesics on meshes, ACM Trans Graph, 2005
[13] Xin SQ., Wang GJ., Improving Chen and Han's algorithm on the discrete geodesic problem, ACM Trans Graph, 2009
[14] Xin SQ., Ying X., He Y., Constant-time all-pairs geodesic distance query on triangle meshes, Proc. ACM SIGGRAPH Symp. Interactive 3D Graph. Games, 2012
[15] Xu C., Wang TY., Liu Y., Liu L., He Y., Fast wavefront propagation (FWP) for computing exact geodesic distances on meshes, IEEE Trans Vis Comput Graphics, 2015
[16] Ying X., Wang X., He Y., Saddle vertex graph (SVG): A novel solution to the discrete geodesic problem, ACM Trans. Graph., 2013
[17] Guzek K., Napieralski P.: Efficient rendering of caustics with streamed photon mapping, Bulletin of the Polish Academy of Sciences, Technical Sciences 65(3): pp. 361-368 · June 2017, ISSN 2300-1917
[18] Olejnik M., Szajerman D., Napieralski P.: Physically Based Area Lighting Model for Real-Time Animation Computer Vision and Graphics, International Conference, ICCVG 2016, Warsaw, Poland, September 19-21, 2016. Proceedings: pp. 73-85