



**HAL**  
open science

# An optimal triangle projector with prescribed area and orientation, application to position-based dynamics

Carlos Arango Duque, Adrien Bartoli

► **To cite this version:**

Carlos Arango Duque, Adrien Bartoli. An optimal triangle projector with prescribed area and orientation, application to position-based dynamics. *Graphical Models*, 2021, 118, pp.101117. 10.1016/j.gmod.2021.101117 . hal-03681222

**HAL Id: hal-03681222**

**<https://uca.hal.science/hal-03681222v1>**

Submitted on 5 Jan 2024

**HAL** is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.



Distributed under a Creative Commons Attribution - NonCommercial 4.0 International License

# An Optimal Triangle Projector with Prescribed Area and Orientation, Application to Position-Based Dynamics

Carlos Arango Duque\*, Adrien Bartoli

*EnCoV, IGT, Institut Pascal, UMR6602 CNRS Université Clermont Auvergne*

---

## Abstract

The vast majority of mesh-based modelling applications iteratively transform the mesh vertices under prescribed geometric conditions. This occurs in particular in methods cycling through the constraint set such as Position-Based Dynamics (PBD). A common case is the approximate local area preservation of triangular 2D meshes under external editing constraints. At the constraint level, this yields the nonconvex optimal triangle projection under prescribed area problem, for which there does not currently exist a direct solution method. In current PBD implementations, the area preservation constraint is linearised. The solution comes out through the iterations, without a guarantee of optimality, and the process may fail for degenerate inputs where the vertices are colinear or colocated. We propose a closed-form solution method and its numerically robust algebraic implementation. Our method handles degenerate inputs through a two-case analysis of the problem's generic ambiguities. We show in a series of experiments in area-based 2D mesh editing that using optimal projection in place of area constraint linearisation in PBD speeds up and stabilises convergence.

*Keywords:* triangle, optimal projection, area preservation, orientation preservation, mesh editing, PBD

---

## 1. Introduction

1 A key mechanism in many mesh-based modelling applications is to transform the  
2 mesh vertices to meet prescribed geometric conditions. For example, triangular mesh  
3 smoothing may be achieved by moving the vertices of each triangle by a specifically  
4

---

\*Corresponding author

*Email address:* `Carlos.ARANGO_DUQUE@uca.fr` (Carlos Arango Duque)

5 designed two-step stretching-shrinking transformation [1] or by iteratively applying a  
6 local smoothing transformation [2]. Another example is 3D volumetric model defor-  
7 mation, where realism is improved by preserving the volume of the mesh’s tetrahe-  
8 drons [3, 4]. Position-Based Dynamics (PBD) is a widely used simulation technique  
9 that directly manipulates the vertex positions of object meshes. It can model various  
10 object behaviours such as rigid body, soft body and fluids [5]. Due to its simplicity,  
11 robustness and speed, PBD has become very popular in computer graphics and in the  
12 video-game industry. In general terms, PBD updates the vertex positions through  
13 simple integration of the external forces. These positions are then directly subjected  
14 to a series of constraint equations handled one at a time. If the obtained projection  
15 minimises the vertex displacement then it is qualified as optimal. For example, the  
16 projection for vertex distance preservation is a simple problem, which was solved  
17 optimally [6, 7]. The constraints simulate a wide range of effects like stretching,  
18 bending, collision, area and volume conservation [7]. Over the years, improvements  
19 have been proposed to the original formulation of PBD. These include new bending  
20 constraints from simple geometric principles [8, 9], stability improvement by geomet-  
21 ric stiffness [10] and faster convergence by constraint reordering [11].

22 Mesh editing uses a priori chosen fixed vertices and moving vertices which should  
23 respect constraints. In 2D triangular mesh editing, local area preservation is a widely  
24 used constraint. The mesh is deformed until the triangle-wise area variation is min-  
25 imised. Enforcing this constraint leads to the Optimal Triangle Projection with  
26 Prescribed Area problem (OTPPA), which we will formally define shortly. OTPPA  
27 is a difficult problem and has not yet been given a closed-form solution in the litera-  
28 ture, contrarily to optimal vertex distance preservation. We formally define OTPPA  
29 as follows. We define the vertices  $v_a$ ,  $v_b$  and  $v_c$  of a triangle in a 2D space as a 6D  
30 vector  $\mathbf{v}$  with:

$$\mathbf{v} = [v_a, v_b, v_c]^\top = [x_a, y_a, x_b, y_b, x_c, y_c]^\top \in \mathbb{R}^6. \quad (1)$$

31 We denote the input triangle  $\tilde{\mathbf{v}} = [\tilde{x}_a, \tilde{y}_a, \tilde{x}_b, \tilde{y}_b, \tilde{x}_c, \tilde{y}_c]^\top$  and the prescribed area  $A_o$ .  
32 We assume  $A_o > 0$  out of practical considerations<sup>1</sup>. The general OTPPA problem is  
33 stated as:

$$\min_{\mathbf{v} \in \mathbb{R}^6} \mathcal{E}(\mathbf{v}) \quad \text{s.t.} \quad f(\mathbf{v}) = 0, \quad (2)$$

34 where  $\mathcal{E}(\mathbf{v})$  is the least-squares *displacement cost*:

$$\mathcal{E}(\mathbf{v}) = \|\mathbf{v} - \tilde{\mathbf{v}}\|^2, \quad (3)$$

---

<sup>1</sup> $A_o = 0$  implies that the resulting vertices are colinear, in which case they are simply given by the orthogonal projection of the input vertices onto a best-fit least-squares line to the input vertices.

35 and  $f(\mathbf{v})$  is the nonconvex *area preservation constraint*, defined from the triangle  
 36 area function  $A(\mathbf{v})$  as:

$$f(\mathbf{v}) = A(\mathbf{v}) - A_o. \quad (4)$$

37 Areas are positive quantities. This means that when we calculate the area of a  
 38 triangle using its vertices, we obtain a positive value regardless of their orientation.  
 39 In this formulation, the area constraint is the difference of two non-negative values.  
 40 That is, the area of the triangle is constrained but not its orientation. For instance,  $\mathbf{v}$   
 41 could be mirrored or two vertices could be swapped and the area constraint would still  
 42 be satisfied. This could result in undesired triangle inversions in mesh editing. We  
 43 thus introduce a related problem that additionally constrains the triangle orientation.  
 44 We first define the triangle area function as  $A(\mathbf{v}) = |A^*(\mathbf{v})|$ , where  $A^*(\mathbf{v})$  is the *signed*  
 45 *area* given by the shoelace formula:

$$A^*(\mathbf{v}) = \frac{(x_a - x_c)(y_b - y_a) - (x_a - x_b)(y_c - y_a)}{2}. \quad (5)$$

46 The signed area is more informative than the area. Specifically,  $\text{sign}(A^*(\mathbf{v}))$  gives  
 47 the triangle orientation. We use this property to define the additional orientation  
 48 constraint. This leads to the Optimal Triangle Projection with Prescribed Area and  
 49 Orientation problem (OTPPAO), stated as:

$$\min_{\mathbf{v} \in \mathbb{R}^6} \mathcal{C}(\mathbf{v}) \quad \text{s.t.} \quad f(\mathbf{v}) = 0 \quad \text{and} \quad g(\mathbf{v}) = 0, \quad (6)$$

50 where  $g(\mathbf{v})$  is the *orientation preservation constraint*:

$$g(\mathbf{v}) = \text{sign}(A^*(\mathbf{v})) - s, \quad (7)$$

51 and  $s \in \{-1, 1\}$  specifies the prescribed orientation. The value chosen for  $s$  depends  
 52 on the application. For instance, in PBD, one would choose the orientation of the  
 53 reference mesh triangle, while another possibility would be to preserve the orientation  
 54 of the input triangle by setting  $s = \text{sign}(A^*(\tilde{\mathbf{v}}))$ .

55 For both OTPPA and OTPPAO, current PBD implementations linearise the area  
 56 preservation constraint, resulting in suboptimal projection. For OTPPAO, they also  
 57 check and enforce the orientation constraint a posteriori in the inner optimisation  
 58 loop. There also exist implementations based on non-linear optimisation, namely Se-  
 59 quential Quadratic Programming (SQP), which may result in faster convergence [12].  
 60 Similarly to linearisation, these methods may fail to find the global minimum. This  
 61 will be critical when the problem is close to a degenerate configuration admitting  
 62 several solutions. In addition, any iterative method will require an unpredictable

63 number of iterations to converge, rendering the actual computation time difficult to  
64 predict. Furthermore, they degenerate for inputs where the vertices are colinear or  
65 colocated, whereas a reliable solution should handle any input. We can thus expect  
66 an optimal projection to improve PBD convergence compared to linearisation and  
67 iterative methods.

68 We propose a closed-form method to OTPPAO and OTPPA. Our method for  
69 OTPPAO handles degenerate inputs through a two-case analysis, guaranteeing it  
70 to find the optimal solution and returning multiple optimal solutions for ambiguous  
71 inputs. Our method for OTPPA directly relies on OTPPAO and shares the same  
72 features. Our method derivation, although thoroughly detailed and relying on some  
73 complex steps, results in a simple algebraic procedure which can be readily imple-  
74 mented in any programming language. We use our closed-form method to implement  
75 PBD, hence dubbed PBD-opt, for mesh editing and compare its performance with  
76 respect to the existing PBD implementation with linearisation, dubbed PBD-lin. To  
77 illustrate our proposal, we present a one triangle toy example in Figure 1 in which we  
78 wish to resize the triangle to half its initial area. PBD-lin takes several iterations to  
79 reach the prescribed area, whereas PBD-opt achieves it directly. The cost evolution  
80 shows that PBD-lin starts with a lower cost, but by the time it complies with the  
81 area constraint, it reaches a larger cost than PBD-opt, indicating convergence to a  
82 local suboptimal minimum.

83 This paper has two parts. In the first part, we derive our closed-form methods.  
84 We show that they deal with generic ambiguities. We then implement our methods  
85 as numerically robust algebraic procedures. In the second part, we embed our alge-  
86 braic procedure for OTPPAO in PBD to form an implementation of PBD-opt. We  
87 compare its performance in convergence speed and stability with respect to the ex-  
88 isting PBD-lin in a series of experiments **and present some use-cases**. We finally give  
89 a complementary section where we present our solution to OTPPA and specialise  
90 our methods to cases where one or two triangle vertices are fixed, which is typically  
91 applicable to triangles at the domain boundary in mesh editing.

## 92 **2. Optimal Triangle Projection with Prescribed Area and Orientation**

93 The derivation of our closed-form method to OTPPAO starts by combining the  
94 two constraints into a single one related to both triangle area and orientation. We  
95 then construct the Lagrangian which leads to a nonconvex problem, which we handle  
96 with two cases. We distinguish and geometrically interpret the two cases based on  
97 the input vertices. In the first case, we reformulate the problem as a depressed  
98 quartic equation and solve it analytically. In the second case, we reformulate the

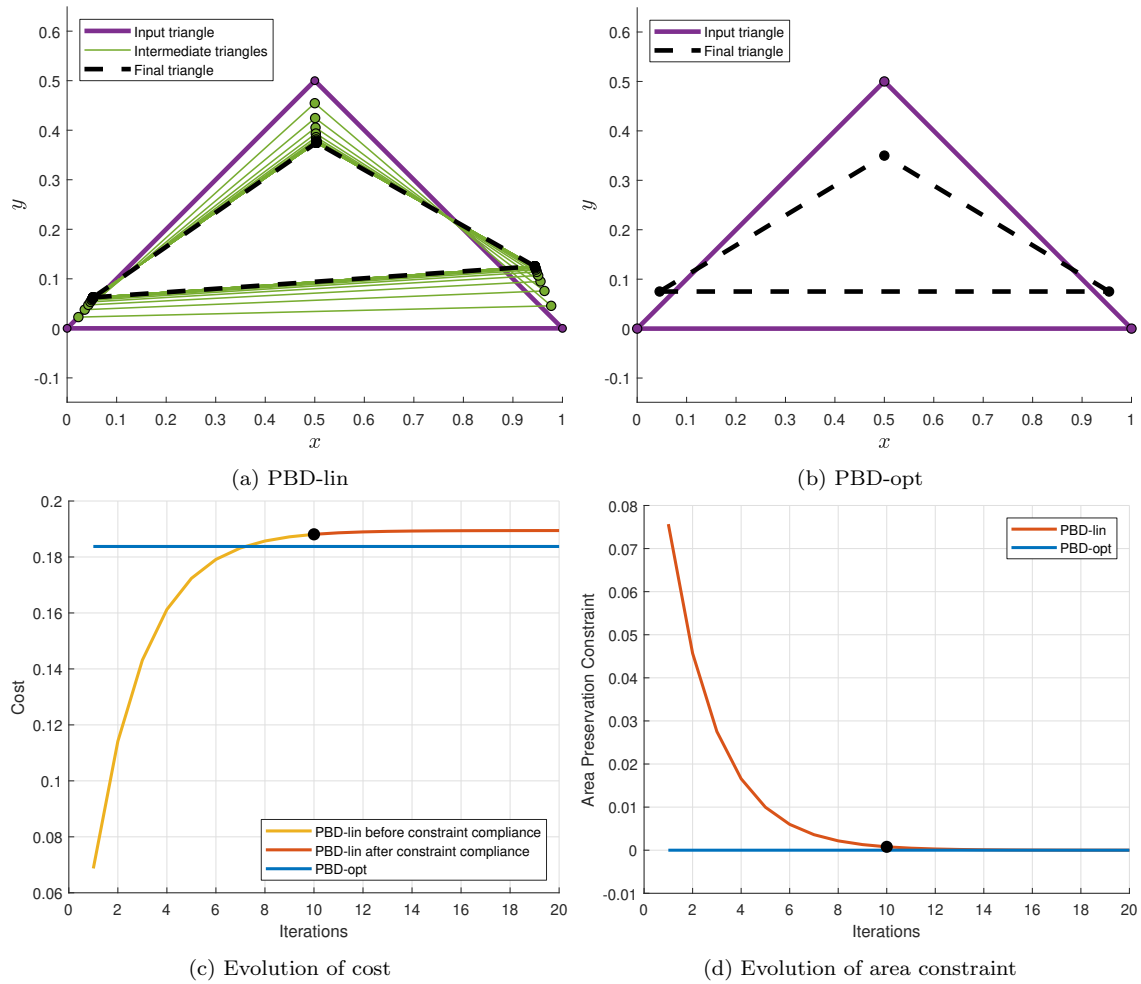


Figure 1: Method comparison on a one triangle toy example. (a), PBD-lin resizes the initial triangle (purple) into smaller intermediary triangles (green) until it reaches a triangle with the prescribed area (black dashed). (b), PBD-opt directly reaches the prescribed area. (c), Comparison of the evolution of the cost of PBD-lin and the fixed cost of PBD-opt. (d), Comparison of the evolution of the prescribed area constraint of PBD-lin and the fixed area of PBD-opt. The cost of PBD-opt (blue) is constant as it gives a direct solution. The cost of PBD-lin is lower during the first iterations (yellow) but the area constraint tolerance  $|A(\mathbf{v}) - A_o| \leq 10^{-3}$  is not yet fulfilled. By the time it reaches the prescribed area (black dot), the cost of PBD-lin (red) has become larger than the one of PBD-opt. Furthermore, after 20 iterations the area constraint for PBD-lin is  $4.802 \cdot 10^{-6}$  compared to  $8.327 \cdot 10^{-17}$  in PBD-opt.

99 problem as a series of homogeneous equations and find its null space. Based on these  
 100 procedures, we develop a numerically robust algebraic implementation and show the  
 101 results in a series of illustrative examples.

### 102 *2.1. Single Constraint Reformulation*

103 We reformulate OTPPAO by merging the area and orientation constraints into a  
 104 single equivalent constraint:

$$f^*(\mathbf{v}) = 0 \quad \text{with} \quad f^*(\mathbf{v}) = sA^*(\mathbf{v}) - A_o. \quad (8)$$

105 We have  $(f(\mathbf{v}) = 0) \wedge (g(\mathbf{v}) = 0) \Leftrightarrow f^*(\mathbf{v}) = 0$ . The forward implication is ob-  
 106 tained by rewriting  $f(\mathbf{v}) = 0$  as  $\text{sign}(A^*(\mathbf{v}))A^*(\mathbf{v}) - A_o = 0$  and substituting  
 107  $s = \text{sign}(A^*(\mathbf{v}))$ , as obtained from  $g(\mathbf{v}) = 0$ , directly giving  $f^*(\mathbf{v}) = 0$ . The re-  
 108 verse implication is obtained by rewriting  $f^*(\mathbf{v}) = 0$  as  $sA^*(\mathbf{v}) = A_o$ , whose absolute  
 109 value gives  $f(\mathbf{v}) = 0$  and whose sign gives  $g(\mathbf{v}) = 0$ . With this new constraint, we  
 110 reformulate OTPPAO as:

$$\min_{\mathbf{v} \in \mathbb{R}^6} \mathcal{C}(\mathbf{v}) \quad \text{s.t.} \quad f^*(\mathbf{v}) = 0. \quad (9)$$

111 This reformulation increases compactness but, more importantly, in contrast to the  
 112 previous area constraint, the new constraint does not involve an absolute value.  
 113 More specifically,  $f^*(\mathbf{v})$  is a nonconvex but smooth function of  $\mathbf{v}$ , meaning that a  
 114 Lagrangian formulation can now be safely constructed.

### 115 *2.2. Lagrangian Formulation*

116 The Lagrangian of the OTPPAO problem (9) is:

$$\mathcal{L}(\mathbf{v}, \lambda) = \mathcal{C}(\mathbf{v}) + \lambda f^*(\mathbf{v}), \quad (10)$$

where  $\lambda$  is the Lagrange multiplier. Setting the gradient to nought we obtain:

$$\frac{\partial \mathcal{L}}{\partial \lambda} = f^*(\mathbf{v}) = sA^*(\mathbf{v}) - A_o = 0 \quad (11)$$

$$\frac{\partial \mathcal{L}}{\partial \mathbf{v}} = \frac{\partial \mathcal{C}(\mathbf{v})}{\partial \mathbf{v}} + \lambda \frac{\partial f^*(\mathbf{v})}{\partial \mathbf{v}} = 2(\mathbf{v} - \tilde{\mathbf{v}}) + s\lambda \frac{\partial A^*(\mathbf{v})}{\partial \mathbf{v}} = 0. \quad (12)$$

Expanding  $\frac{\partial \mathcal{L}}{\partial \mathbf{v}}$ , we obtain the following six equations:

$$\begin{aligned}\frac{\partial \mathcal{L}}{\partial x_a} &= 2(x_a - \tilde{x}_a) + s\frac{\lambda}{2}(y_b - y_c) = 0 \\ \frac{\partial \mathcal{L}}{\partial y_a} &= 2(y_a - \tilde{y}_a) + s\frac{\lambda}{2}(x_c - x_b) = 0 \\ \frac{\partial \mathcal{L}}{\partial x_b} &= 2(x_b - \tilde{x}_b) + s\frac{\lambda}{2}(y_c - y_a) = 0 \\ \frac{\partial \mathcal{L}}{\partial y_b} &= 2(y_b - \tilde{y}_b) + s\frac{\lambda}{2}(x_a - x_c) = 0 \\ \frac{\partial \mathcal{L}}{\partial x_c} &= 2(x_c - \tilde{x}_c) + s\frac{\lambda}{2}(y_a - y_b) = 0 \\ \frac{\partial \mathcal{L}}{\partial y_c} &= 2(y_c - \tilde{y}_c) + s\frac{\lambda}{2}(x_b - x_a) = 0.\end{aligned}$$

117 We rewrite these equations in matrix form as:

$$X\mathbf{v} = \tilde{\mathbf{v}}, \quad (13)$$

118 where  $X \in \mathbb{R}^{6 \times 6}$  is given by:

$$X = \begin{bmatrix} 1 & 0 & 0 & s\lambda/4 & 0 & -s\lambda/4 \\ 0 & 1 & -s\lambda/4 & 0 & s\lambda/4 & 0 \\ 0 & -s\lambda/4 & 1 & 0 & 0 & s\lambda/4 \\ s\lambda/4 & 0 & 0 & 1 & -s\lambda/4 & 0 \\ 0 & s\lambda/4 & 0 & -s\lambda/4 & 1 & 0 \\ -s\lambda/4 & 0 & s\lambda/4 & 0 & 0 & 1 \end{bmatrix}. \quad (14)$$

119 *2.3. Solving with Two Cases*

120 We want to solve for  $\mathbf{v}$  from equation (14). We first check the invertibility of  $X$   
121 from its determinant:

$$\det(X) = \frac{(3\lambda^2 - 16)^2}{256}. \quad (15)$$

122 We thus have:

$$\det(X) = 0 \quad \Leftrightarrow \quad |\lambda| = \lambda_o, \quad (16)$$

123 where  $\lambda_o$  correspond to the reciprocal area of a normalised equilateral triangle:

$$\lambda_o = \frac{4}{\sqrt{3}}. \quad (17)$$



124 We show in the next section that this special case is related to input vertices repre-  
 125 senting an equilateral triangle or being colocated. We thus solve system (14) with  
 126 two cases. In Case I, which is the most general one, we have  $|\lambda| \neq \lambda_o$ . In Case II,  
 127 we have  $|\lambda| = \lambda_o$ .

#### 128 2.4. Geometrically Interpreting and Distinguishing the Two Cases

129 The problem setting is defined by the input vertices  $\tilde{\mathbf{v}}$ , the prescribed area  $A_o$   
 130 and orientation  $s$ . Most settings are solved by Case I and exceptions are handled  
 131 by Case II. Both cases can be geometrically interpreted and distinguished from each  
 132 other based on three criteria:

- 133 • **Linear deficiency of  $\tilde{\mathbf{v}}$ .** This is evaluated as the rank of matrix  $M \in \mathbb{R}^{3 \times 3}$   
 134 containing  $\tilde{\mathbf{v}}$  in homogeneous coordinates as:

$$M \stackrel{\text{def}}{=} \begin{bmatrix} \tilde{x}_a & \tilde{y}_a & 1 \\ \tilde{x}_b & \tilde{y}_b & 1 \\ \tilde{x}_c & \tilde{y}_c & 1 \end{bmatrix}. \quad (18)$$

135 For most configurations  $\text{rank}(M) = 3$ , which means that the vertices are not  
 136 aligned and represent any given triangle whose area  $A(\tilde{\mathbf{v}})$  is non-zero. In con-  
 137 trast,  $\text{rank}(M) = 2$  means that the three vertices are colinear, in which case  
 138  $A(\tilde{\mathbf{v}}) = 0$ . Finally,  $\text{rank}(M) = 1$  means that the three vertices are colocated  
 139 and also implies  $A(\tilde{\mathbf{v}}) = 0$ .

- 140 • **Orientation change of  $\tilde{\mathbf{v}}$ .** This is evaluated by comparing the input vertices  
 141 orientation  $\text{sign}(A^*(\tilde{\mathbf{v}}))$  and the prescribed orientation  $s$ . When the orientation  
 142 of the input vertices is preserved then  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ . On the other hand,  
 143 when the orientation of the input vertices is inverted then  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$ .
- 144 • **Scale of  $A(\tilde{\mathbf{v}})$  with respect to  $A_o$ .** This is only useful in the special case  
 145 of an equilateral triangle with preserved orientation. It refers to whether the  
 146 absolute value of the scaled input area  $|zA^*(\tilde{\mathbf{v}})|$  is larger than, equal to or  
 147 smaller than the prescribed area  $A_o$  for some  $z \in \mathbb{R} > 0$ .

148 The interpretation of Cases I and II with the above criteria is given in Table 1  
 149 and summarised by the following Proposition.

150 **Proposition 1.** *We define a problem setting as the input vertices  $\tilde{\mathbf{v}}$ , the prescribed*  
 151 *area  $A_o$  and orientation  $s$ . Most settings fall in Case I and are then denoted  $S_o$ .*  
 152 *Exceptions, handled with Case II, are:*

- 153 •  $S_1$ :  $\tilde{\mathbf{v}}$  is a single point
- 154 •  $S_2$ :  $\tilde{\mathbf{v}}$  is an equilateral triangle and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$
- 155 •  $S_3$ :  $\tilde{\mathbf{v}}$  is an equilateral triangle,  $A(\tilde{\mathbf{v}})/4 \geq A_o$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$

156 The proof of Proposition 1 is based on the following five lemmas.

157 **Lemma 1.**  $S_1 \iff A(\tilde{\mathbf{v}}) = 0$  and  $|\lambda| = \lambda_o$ .

158 **Lemma 2.**  $S_2 \iff A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = -\lambda_o$ .

159 **Lemma 3.**  $S_3 \Rightarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda \in \left\{ \lambda_o, -\lambda_o + \sqrt{\frac{\lambda_o}{A_o}}, -\lambda_o - \sqrt{\frac{\lambda_o}{A_o}} \right\}$ .

160 **Lemma 4.**  $S_3 \Leftarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = \lambda_o$ .

161 **Lemma 5.** Choosing  $\lambda = \lambda_o$  leads to the optimal solution for  $S_3$ .

162 The proofs of these lemmas are given in Appendix A.

163 *Proof of Proposition 1.* We recall that Case I occurs for  $|\lambda| \neq \lambda_o$  and Case II for  
 164  $|\lambda| = \lambda_o$ . Lemmas 1, 2 and 4 show that  $S_1$ ,  $S_2$  and  $S_3$  are the only possible settings  
 165 corresponding to  $|\lambda| = \lambda_o$ , hence possibly to Case II. This proves that Case I is the  
 166 general case. Lemmas 1 and 2 then trivially prove that  $S_1$  and  $S_2$  are handled by  
 167 Case II. Finally, lemmas 3 and 5 prove that  $S_3$  is also handled by Case II.  $\square$

### 168 2.5. Case I

169 Case I is the most general one. It occurs for  $|\lambda| \neq \lambda_o$ , equivalent to  $\det(X) \neq 0$ .  
 170 From Proposition 1, we have  $\text{rank}(M) \geq 2$ , in other words, at least one of the initial  
 171 vertices  $\tilde{\mathbf{v}}$  is different from the other two (except if the input is an equilateral triangle  
 172 under the conditions of Proposition 1). We follow two steps. We first eliminate the  
 173 vertices from the equations, which leads to a depressed quartic in  $\lambda$ . We then find  
 174 the roots of this quartic using Ferrari's method and trivially solve for the vertices  
 175 from the initial linear system (13).

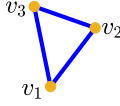
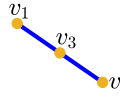

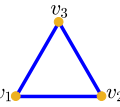
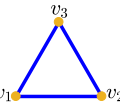

Case	Setting	Input	$\det(X)$	$\text{rank}(M)$	$A(\tilde{\mathbf{v}})$	$\sigma^2(\tilde{\mathbf{v}})$	Number of solutions	$\text{sign}(A(\tilde{\mathbf{v}}))$
I	$S_o$		non zero	3	non zero	non zero	$\leq 4$	$\pm 1$
			zero	2	zero	zero	$\infty$	-1
II	$S_1$	$v_1=v_2=v_3$ 	zero	1	zero	zero	$\infty$	$\pm 1$
I	$S_o$		non zero		$A(\tilde{\mathbf{v}})/4 \leq A_o$	non zero	$\leq 4$	1
II	$S_3$		zero	3	$A(\tilde{\mathbf{v}})/4 \geq A_o$	non zero	$\infty$	1
II	$S_2$		zero		non zero	non zero	$\infty$	-1

Table 1: Characteristics of Cases I and II and number of solutions.

176 *2.5.1. Polynomial Reformulation*

177 Our reformulation proceeds by expressing the vertices in  $\tilde{\mathbf{v}}$  as a function of  $\lambda$   
 178 scaled by the determinant and substituting in the signed area constraint  $f^*(\mathbf{v}) = 0$ .  
 179 We start by multiplying equation (13) by the adjugate  $X^*$  of  $X$  and obtain:

$$\det(X)\mathbf{v} = X^*\tilde{\mathbf{v}}, \quad (19)$$

180 where the adjugate is:

$$X^* = \frac{\delta}{256}Y = \frac{3\lambda^2 - 16}{256} \begin{bmatrix} \lambda^2 - 16 & 0 & \lambda^2 & 4s\lambda & \lambda^2 & -4s\lambda \\ 0 & \lambda^2 - 16 & -4s\lambda & \lambda^2 & 4s\lambda & \lambda^2 \\ \lambda^2 & -4s\lambda & \lambda^2 - 16 & 0 & \lambda^2 & 4s\lambda \\ 4s\lambda & \lambda^2 & 0 & \lambda^2 - 16 & -4s\lambda & \lambda^2 \\ \lambda^2 & 4s\lambda & \lambda^2 & -4s\lambda & \lambda^2 - 16 & 0 \\ -4s\lambda & \lambda^2 & 4s\lambda & \lambda^2 & 0 & \lambda^2 - 16 \end{bmatrix}, \quad (20)$$

181 with  $\delta = 3\lambda^2 - 16$  and  $Y \in \mathbb{R}^{6 \times 6}$ . We notice the following:

$$\det(X) = \frac{\delta^2}{256}. \quad (21)$$

182 We substitute equations (20) and (21) in equation (19) and obtain:

$$\delta\mathbf{v} = Y\tilde{\mathbf{v}}. \quad (22)$$

183 We observe that the signed area  $A^*(\delta\mathbf{v}) = \delta^2 A^*(\mathbf{v})$ . Thus, we calculate the signed  
 184 area of both sides of equation (19) and obtain:

$$\delta^2 A^*(\mathbf{v}) = A^*(Y\tilde{\mathbf{v}}). \quad (23)$$

185 After some minor manipulations, we obtain:

$$A^*(Y\tilde{\mathbf{v}}) = a_2\lambda^2 + a_1\lambda + a_0, \quad (24)$$

where:

$$\begin{aligned} a_0 &= 128((\tilde{x}_a - \tilde{x}_c)(\tilde{y}_b - \tilde{y}_a) - (\tilde{x}_a - \tilde{x}_b)(\tilde{y}_c - \tilde{y}_a)) \\ a_1 &= -64s(\tilde{x}_a^2 + \tilde{x}_b^2 + \tilde{x}_c^2 + \tilde{y}_a^2 + \tilde{y}_b^2 + \tilde{y}_c^2 - \tilde{x}_a\tilde{x}_b - \tilde{x}_a\tilde{x}_c - \tilde{x}_b\tilde{x}_c - \tilde{y}_a\tilde{y}_b - \tilde{y}_a\tilde{y}_c - \tilde{y}_b\tilde{y}_c) \\ a_2 &= 24((\tilde{x}_a - \tilde{x}_c)(\tilde{y}_b - \tilde{y}_a) - (\tilde{x}_a - \tilde{x}_b)(\tilde{y}_c - \tilde{y}_a)). \end{aligned}$$

186 We can rewrite these coefficients more compactly. Concretely,  $a_0$  and  $a_2$  contain the  
 187 signed area of the input vertices  $A^*(\tilde{\mathbf{v}})$ , as given by equation (5). Furthermore,  $a_1$

188 is proportional to the sum of the variance of the  $x$  and  $y$  components of the input  
 189 vertices, as  $\sigma^2(\tilde{\mathbf{v}}) = \sigma_x^2(\tilde{\mathbf{v}}) + \sigma_y^2(\tilde{\mathbf{v}})$  where:

$$\sigma_x^2(\tilde{\mathbf{v}}) = \frac{(\tilde{x}_a - \bar{x})^2 + (\tilde{x}_b - \bar{x})^2 + (\tilde{x}_c - \bar{x})^2}{3}, \quad (25)$$

190

$$\sigma_y^2(\tilde{\mathbf{v}}) = \frac{(\tilde{y}_a - \bar{y})^2 + (\tilde{y}_b - \bar{y})^2 + (\tilde{y}_c - \bar{y})^2}{3}, \quad (26)$$

191 and  $\bar{x}$  and  $\bar{y}$  are the  $x$  and  $y$  components of the input triangle's centroid. We then  
 192 have:

$$\frac{3}{2}\sigma^2(\tilde{\mathbf{v}}) = \tilde{x}_a^2 + \tilde{x}_b^2 + \tilde{x}_c^2 + \tilde{y}_a^2 + \tilde{y}_b^2 + \tilde{y}_c^2 - \tilde{x}_a\tilde{x}_b - \tilde{x}_a\tilde{x}_c - \tilde{x}_b\tilde{x}_c - \tilde{y}_a\tilde{y}_b - \tilde{y}_a\tilde{y}_c - \tilde{y}_b\tilde{y}_c, \quad (27)$$

193 and thus,  $a_1 = -96s\sigma^2(\tilde{\mathbf{v}})$ . We substitute equation (23) in the signed area con-  
 194 straint (8) multiplied by  $\delta^2$  and obtain:

$$sA^*(Y\tilde{\mathbf{v}}) - \delta^2 A_o = 0. \quad (28)$$

195 This way, the signed area only depends on the known initial vertices  $\tilde{\mathbf{v}}$  and prescribed  
 196 sign  $s$ . Because the signed area is quadratic in the vertices, and the vertices are  
 197 quadratic rational in  $\lambda$ , the resulting equation is a quartic in  $\lambda$ :

$$9A_o\lambda^4 - 48(2A_o + sA^*(\tilde{\mathbf{v}}))\lambda^2 + 96\sigma^2(\tilde{\mathbf{v}})\lambda + 256(A_o - sA^*(\tilde{\mathbf{v}})) = 0. \quad (29)$$

198 This is a depressed quartic because it does not have a cubic term. We can thus  
 199 rewrite it to the standard form by simply dividing by  $9A_o$ , giving:

$$\lambda^4 + p\lambda^2 + q\lambda + r = 0, \quad (30)$$

with:

$$p = -\frac{16(2A_o + sA^*(\tilde{\mathbf{v}}))}{3A_o} \quad (31)$$

$$q = \frac{32\sigma^2(\tilde{\mathbf{v}})}{3A_o} \quad (32)$$

$$r = \frac{256(A_o - sA^*(\tilde{\mathbf{v}}))}{9A_o}. \quad (33)$$

200 An important question is whether we can further simplify the depressed quartic by  
 201 nullifying one of its coefficients. The possible actions lie in choosing the coordinate

202 frame in which the vertices are expressed using a proper scaled Euclidean transforma-  
 203 tion. Coefficients  $p$  and  $r$  are linear combinations of the triangle areas  $A^*(\tilde{\mathbf{v}})$  and  $A_o$   
 204 and are normalised by  $A_o$ . This means that they are scale, rotation and translation  
 205 invariant, and thus cannot be cancelled. Coefficient  $q$  is proportional to the variance  
 206  $\sigma^2(\tilde{\mathbf{v}})$  which is also rotation and translation invariant, and thus cannot be cancelled.  
 207 Because it is normalised by the area, it is also scale invariant. **Consequently, since**  
 208  **$\lambda$  is a root of this polynomial, it is also rotation, translation and scale invariant.**  
 209 Therefore, the depressed quartic cannot be simplified. The next step is to solve the  
 210 depressed quartic.

### 211 2.5.2. Solution using Ferrari's Method

212 We have chosen Ferrari's method [13, 14] to solve the quartic equation<sup>2</sup>. The  
 213 method details and proof may be found in Appendix C. We here give its main  
 214 steps for the sake of completeness and for the construction of our numerically robust  
 215 procedure in section 2.8. We first extract the resolvent cubic for equation (30). We  
 216 then use Cardano's formula to extract the real root  $\alpha_o$  of the resolvent cubic as:

$$\alpha_o = \sqrt[3]{Q_2 + \sqrt{Q_1^3 + Q_2^2}} + \sqrt[3]{Q_2 - \sqrt{Q_1^3 + Q_2^2}} - \frac{p}{3} \quad (34)$$

where:

$$Q_1 = -\frac{p^2 + 12r}{36} \quad (35)$$

$$Q_2 = \frac{2p^3 - 72rp + 27q^2}{432}. \quad (36)$$

217 The expansion of  $Q_1$  and  $Q_2$  does not bring simplified expressions. We note that  
 218 Proposition 1 implies  $q \neq 0$ , thus  $\alpha_o \neq 0$ . We finally use  $\alpha_o$  to extract the roots of  
 219 the depressed quartic as:

$$\lambda = \frac{s_1\sqrt{\alpha_o} + s_2\sqrt{-\left(p + \alpha_o + s_1\frac{q}{\sqrt{2\alpha_o}}\right)}}{\sqrt{2}}, \quad (37)$$

220 where  $s_1, s_2 \in \{-1, 1\}$ , leaving four possibilities, hence four roots. Substituting  
 221 these roots in equation (13), we obtain four sets of vertices, at least one of which  
 222 representing an optimal solution to OTPPAO.

---

<sup>2</sup>There are five main types of solution methods for a quartic equation. There does not seem to exist a consensus as to which one should be preferred in terms of stability [15, 16, 17].

223 *2.6. Case II*

224 Case II is a special case. It occurs for  $|\lambda| = \lambda_o$ , equivalent to  $\det(X) = 0$ .  
 225 From Proposition 1, this means that the initial vertices  $\tilde{\mathbf{v}}$  either are colocated as  
 226  $\tilde{v}_a = \tilde{v}_b = \tilde{v}_c$  or represent equilateral triangles under the conditions of Proposition 1.  
 227 We show that the problem is represented by translated homogeneous and linearly  
 228 dependent equations. We find their null space and then a subset constrained by the  
 229 prescribed area.

230 We translate the coordinate system to bring the input triangle's centroid to the  
 231 origin as  $\tilde{\mathbf{v}}' = \tilde{\mathbf{v}} - \bar{\mathbf{v}}$ , which also translates the unknown vertices to  $\mathbf{v}' = \mathbf{v} - \bar{\mathbf{v}}$ .  
 232 Substituting  $|\lambda| = \lambda_o$  in matrix  $X$ , we obtain:

$$X = \begin{bmatrix} 1 & 0 & 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} \\ 0 & 1 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 \\ 0 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 1 & 0 & 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} \\ \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & 0 & 1 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 \\ 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 1 & 0 \\ -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & 0 & 1 \end{bmatrix}. \quad (38)$$

233 We have  $\det(X) = 0$ , as expected, independently of  $\operatorname{sign}(\lambda)$ . In addition, all  $5 \times 5$   
 234 minors of  $X$  are zero and the leading  $4 \times 4$  minor is non-zero:

$$\det \left( \begin{bmatrix} 1 & 0 & 0 & \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} \\ 0 & 1 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 \\ 0 & -\frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 1 & 0 \\ \frac{s \operatorname{sign}(\lambda)}{\sqrt{3}} & 0 & 0 & 1 \end{bmatrix} \right) = \frac{4}{9}. \quad (39)$$

235 This means that  $\operatorname{rank}(X) = 4$ . Thus,  $X\mathbf{v}' = \tilde{\mathbf{v}}'$  is solvable if and only if  $\tilde{\mathbf{v}}'$  lies in  
 236 the column space  $C(X)$ . The column space can be calculated by factoring  $X$  into  
 237 its Singular Value Decomposition (SVD)  $X = U\Sigma U^\top$  ( $X$  is symmetric) and taking  
 238 the first  $\operatorname{rank}(X)$  columns of the unitary matrix  $U$ . For each value of  $s \operatorname{sign}(\lambda)$ , we  
 239 have column spaces expressed as four-dimensional linear subspaces  $\{\gamma_1 \mathbf{u}_1^- + \gamma_2 \mathbf{u}_2^- +$   
 240  $\gamma_3 \mathbf{u}_3^- + \gamma_4 \mathbf{u}_4^-\}$  and  $\{\gamma_1 \mathbf{u}_1^+ + \gamma_2 \mathbf{u}_2^+ + \gamma_3 \mathbf{u}_3^+ + \gamma_4 \mathbf{u}_4^+\}$  where  $\gamma_1, \gamma_2, \gamma_3, \gamma_4 \in \mathbb{R}$  and with

241 bases  $\mathbf{u}_1^-, \mathbf{u}_2^-, \mathbf{u}_3^-, \mathbf{u}_4^- \in \mathbb{R}^6$  and  $\mathbf{u}_1^+, \mathbf{u}_2^+, \mathbf{u}_3^+, \mathbf{u}_4^+ \in \mathbb{R}^6$  such that:

$$242 \quad [\mathbf{u}_1^- \quad \mathbf{u}_2^- \quad \mathbf{u}_3^- \quad \mathbf{u}_4^-] = \begin{bmatrix} 0 & \lambda_o/4 & \lambda_o/4 & 0 \\ \lambda_o/4 & 0 & 0 & \lambda_o/4 \\ 1/2 & -\lambda_o/8 & \lambda_o/4 & 0 \\ -\lambda_o/8 & -1/2 & 0 & \lambda_o/4 \\ -1/2 & -\lambda_o/8 & \lambda_o/4 & 0 \\ -\lambda_o/8 & 1/2 & 0 & \lambda_o/4 \end{bmatrix}, \quad (40)$$

242 and:

$$243 \quad [\mathbf{u}_1^+ \quad \mathbf{u}_2^+ \quad \mathbf{u}_3^+ \quad \mathbf{u}_4^+] = \begin{bmatrix} \lambda_o/4 & 0 & 0 & \lambda_o/4 \\ 0 & \lambda_o/4 & \lambda_o/4 & 0 \\ -\lambda_o/8 & -1/2 & 0 & \lambda_o/4 \\ 1/2 & -\lambda_o/8 & \lambda_o/4 & 0 \\ -\lambda_o/8 & 1/2 & 0 & \lambda_o/4 \\ -1/2 & -\lambda_o/8 & \lambda_o/4 & 0 \end{bmatrix}. \quad (41)$$

243 We have that  $\mathbf{u}_1^-, \mathbf{u}_2^-, \mathbf{u}_1^+$  and  $\mathbf{u}_2^+$  represent centred equilateral triangles of the same  
 244 area of  $\frac{1}{\lambda_o}$  with orientation  $s \operatorname{sign}(\lambda)$  and we have that  $\mathbf{u}_3^-, \mathbf{u}_4^-, \mathbf{u}_3^+$  and  $\mathbf{u}_4^+$  represent  
 245 sets of colocated points. The linear combinations  $\gamma_1 \mathbf{u}_1^- + \gamma_2 \mathbf{u}_2^-$  and  $\gamma_1 \mathbf{u}_1^+ + \gamma_2 \mathbf{u}_2^+$   
 246 represent equilateral triangles of any area and opposite orientations (or colocated  
 247 points if  $\gamma_1 = \gamma_2 = 0$ ), whilst  $\gamma_3 \mathbf{u}_3^- + \gamma_4 \mathbf{u}_4^-$  and  $\gamma_3 \mathbf{u}_3^+ + \gamma_4 \mathbf{u}_4^+$  represent colocated  
 248 points, hence act as a translation for the vertices of the previous linear combination.  
 249 This shows that the system is solvable if and only if  $\tilde{\mathbf{v}}'$  represents an equilateral  
 250 triangle of orientation  $\operatorname{sign}(A^*(\tilde{\mathbf{v}}')) = s \operatorname{sign}(\lambda)$  or colocated vertices.

251 The system  $X \mathbf{v}' = \tilde{\mathbf{v}}'$  is solved by first finding the solutions of the homogeneous  
 252 system  $X \mathbf{v}_h = 0$  and translating them by a particular solution  $\mathbf{v}_p$ , obtaining  $\mathbf{v}' = \mathbf{v}_h +$   
 253  $\mathbf{v}_p$ . The homogeneous system has an infinite number of solutions which come from  
 254 the null space of  $X$ . This can be represented as a two-dimensional linear subspace  
 255  $\mathbf{v}_h = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2$  where the coefficients  $\beta_1, \beta_2 \in \mathbb{R}$  and with bases  $\mathbf{v}_1, \mathbf{v}_2 \in \mathbb{R}^6$  such  
 256 that:

$$257 \quad [\mathbf{v}_1 \quad \mathbf{v}_2] = \begin{bmatrix} -\frac{1}{2} & \frac{2s \operatorname{sign}(\lambda)}{\lambda_o} \\ -\frac{2s \operatorname{sign}(\lambda)}{\lambda_o} & -\frac{1}{2} \\ \frac{\lambda_o}{2} & -\frac{2s \operatorname{sign}(\lambda)}{\lambda_o} \\ \frac{2s \operatorname{sign}(\lambda)}{\lambda_o} & -\frac{1}{2} \\ 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (42)$$

257 We have that  $\mathbf{v}_1, \mathbf{v}_2$  represent centred equilateral triangles of the same area of  $\frac{3}{\lambda_o}$ .  
 258 The linear combination  $\mathbf{v}_h = \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2$  generates centred equilateral triangles of



259 any area of orientation  $-s \operatorname{sign}(\lambda)$ . We then calculate the particular solution  $\mathbf{v}_p$  using  
 260 the pseudo-inverse as:

$$\mathbf{v}_p = X^\dagger \tilde{\mathbf{v}}' = \begin{bmatrix} \frac{\tilde{x}'_a}{2} + \frac{\tilde{x}'_b}{4} + \frac{\tilde{x}'_c}{4} + \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{y}'_b - \tilde{y}'_c}{3\lambda_o} \\ \frac{\tilde{y}'_a}{2} + \frac{\tilde{y}'_b}{4} + \frac{\tilde{y}'_c}{4} - \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{x}'_b - \tilde{x}'_c}{3\lambda_o} \\ \frac{\tilde{x}'_a}{4} + \frac{\tilde{x}'_b}{2} + \frac{\tilde{x}'_c}{4} - \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{y}'_a - \tilde{y}'_c}{3\lambda_o} \\ \frac{\tilde{y}'_a}{4} + \frac{\tilde{y}'_b}{2} + \frac{\tilde{y}'_c}{4} + \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{x}'_a - \tilde{x}'_c}{3\lambda_o} \\ \frac{\tilde{x}'_a}{4} + \frac{\tilde{x}'_b}{4} + \frac{\tilde{x}'_c}{2} + \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{y}'_a - \tilde{y}'_b}{3\lambda_o} \\ \frac{\tilde{y}'_a}{4} + \frac{\tilde{y}'_b}{4} + \frac{\tilde{y}'_c}{2} - \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \frac{\tilde{x}'_a - \tilde{x}'_b}{3\lambda_o} \end{bmatrix}. \quad (43)$$

261 We then translate the null space with the particular solution and obtain  $\mathbf{v}' =$   
 262  $\beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2 + \mathbf{v}_p$ . This linear combination generates centred triangles of any area.  
 263 A noticeable property is stated in the following lemma, whose proof is given in Ap-  
 264 pendix A.

265 **Lemma 6.**  $\mathbf{v}_p = 0 \Rightarrow \mathbf{v}'$  is an equilateral triangle. The converse is not true.

266 The next step is to constrain these triangles to the prescribed area and orientation.  
 267 After some minor algebraic manipulations, we obtain the signed area of the subspace  
 268 as:

$$A^*(\mathbf{v}') = (1 + s \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) \operatorname{sign}(\lambda)) \frac{A^*(\tilde{\mathbf{v}}')}{8} - s \operatorname{sign}(\lambda) \frac{3(\beta_1^2 + \beta_2^2)}{\lambda_o}. \quad (44)$$

269 When  $A^*(\tilde{\mathbf{v}}') \neq 0$  we have  $s \operatorname{sign}(\lambda) = \operatorname{sign}(A^*(\tilde{\mathbf{v}}'))$  thus  $\operatorname{sign}(\lambda) = s \operatorname{sign}(A^*(\tilde{\mathbf{v}}'))$ .  
 270 However, when  $A^*(\tilde{\mathbf{v}}') = 0$  we have  $\operatorname{sign}(A^*(\mathbf{v}')) = -s \operatorname{sign}(\lambda)$ , which implies that  
 271  $\operatorname{sign}(\lambda) = -s$ . Using the orientation constraint (7), we can express  $\mathbf{v}'$  as:

$$\begin{aligned} \mathbf{v}' &= \beta_1 \mathbf{v}_1 + \beta_2 \mathbf{v}_2 + \mathbf{v}_p \\ \text{s.t.} \quad & (s + \operatorname{sign}(\lambda) \operatorname{sign}(A^*(\tilde{\mathbf{v}}'))) \frac{A^*(\tilde{\mathbf{v}}')}{8} - \operatorname{sign}(\lambda) \frac{3(\beta_1^2 + \beta_2^2)}{\lambda_o} - A_o = 0. \end{aligned} \quad (45)$$

272 Because of the area and orientation constraints, and because  $\mathbf{v}_1$  and  $\mathbf{v}_2$  are rotated  
 273 copies of each other, the family defined by equation (42) can be generated by scaling  
 274  $\mathbf{v}_1$  by:

$$\phi = \sqrt{\beta_1^2 + \beta_2^2} = \sqrt{\frac{\lambda_o (\operatorname{sign}(A^*(\tilde{\mathbf{v}}')) A^*(\tilde{\mathbf{v}}') - 4k A_o)}{12}}, \quad (46)$$

275 where  $k$  depends on the type of input:

$$k = \begin{cases} s \operatorname{sign}(A^*(\tilde{\mathbf{v}}')) & \text{if } A^*(\tilde{\mathbf{v}}) \neq 0 \\ -1 & \text{if } A^*(\tilde{\mathbf{v}}) = 0, \end{cases} \quad (47)$$

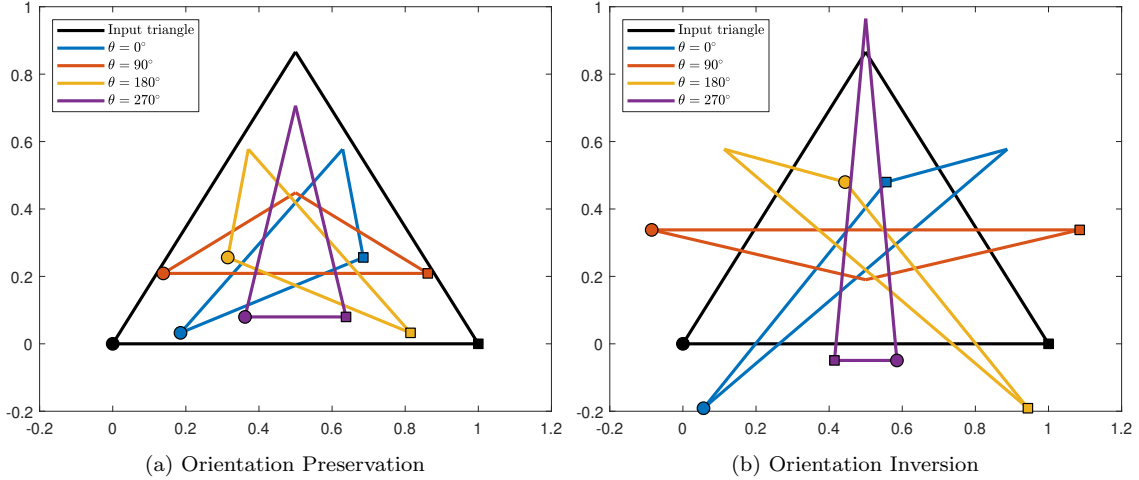


Figure 2: Case II solutions for an equilateral triangle. (a), represents solutions where the input orientation is preserved and (b), where it is inverted. The prescribed area  $A_o$  is  $1/5$  of the input triangle’s area. For both (a) and (b), four solutions corresponding to different values for  $\theta$  are shown and assigned a colour for visual representation. The triangle family is generated by a free 2D rotation followed by a fixed translation in the 6D triangle space, which result in triangles that are not rotated copies of each other. Nevertheless, they all comply with the area preservation constraint and have the same cost.

276 so that the area constraint is met. We can then rotate  $\mathbf{v}_1$  by some arbitrary angle  
 277  $\theta$ . We note that when  $k = s \text{sign}(A^*(\tilde{\mathbf{v}}')) = 1$ , then  $\phi \in \mathbb{R}$  as long as  $A(\tilde{\mathbf{v}})/4 \geq A_o$   
 278 (which corresponds to setting  $S_3$ ). We define a new basis vector  $\mathbf{v}_c$  as:

$$\mathbf{v}_c = \phi \left[ -\frac{1}{2} \quad -ks\frac{2}{\lambda_o} \quad -\frac{1}{2} \quad ks\frac{2}{\lambda_o} \quad 1 \quad 0 \right]^\top. \quad (48)$$

279 We translate it to the original coordinates by  $\mathbf{v}_t$ , which is the addition of the partic-  
 280 ular solution  $\mathbf{v}_p$  and the input’s centroid  $\bar{\mathbf{v}}$ , and obtain:

$$\mathbf{v} = \mathcal{R}(\theta)\mathbf{v}_c + \mathbf{v}_t, \quad (49)$$

281 where  $\mathcal{R}(\theta)$  is a block diagonal matrix replicating the 2D rotation matrix  $R(\theta)$  three  
 282 times as  $\mathcal{R}(\theta) = \text{diag}(R(\theta), R(\theta), R(\theta))$ . Equation (49) generates an infinite num-  
 283 ber of solutions with an identical cost, as shown by the following lemma, proved  
 284 in Appendix A.

285 **Lemma 7.** *All solutions generated by equation (49) have the same cost.*

286 We illustrate some solutions in Case II with a toy example in Figure 2.

287 *2.7. Properties of the Solutions*

An important property of the solutions to OTPPAO is that they preserve the centroid of the input triangle  $\tilde{\mathbf{v}}$ . For Case I, this is shown by substituting the vertices  $\mathbf{v}$  from equation (22) in the centroid formula as:

$$\begin{aligned} \delta \bar{\mathbf{v}} &= \frac{\delta}{3} \begin{bmatrix} x_a + x_b + x_c \\ y_a + y_b + y_c \end{bmatrix} = \frac{1}{3} \begin{bmatrix} (\lambda^2 - 16)(\tilde{x}_a + \tilde{x}_b + \tilde{x}_c) + 2\lambda^2(\tilde{x}_a + \tilde{x}_b + \tilde{x}_c) \\ (\lambda^2 - 16)(\tilde{y}_a + \tilde{y}_b + \tilde{y}_c) + 2\lambda^2(\tilde{y}_a + \tilde{y}_b + \tilde{y}_c) \\ + 4s\lambda(\tilde{y}_b - \tilde{y}_c + \tilde{y}_c - \tilde{y}_a + \tilde{y}_a - \tilde{y}_b) \\ + 4s\lambda(\tilde{x}_b - \tilde{x}_c + \tilde{x}_c - \tilde{x}_a + \tilde{x}_a - \tilde{x}_b) \end{bmatrix} \\ &= \frac{1}{3} \begin{bmatrix} (3\lambda^2 - 16)(\tilde{x}_a + \tilde{x}_b + \tilde{x}_c) \\ (3\lambda^2 - 16)(\tilde{y}_a + \tilde{y}_b + \tilde{y}_c) \end{bmatrix} = \frac{\delta}{3} \begin{bmatrix} \tilde{x}_a + \tilde{x}_b + \tilde{x}_c \\ \tilde{y}_a + \tilde{y}_b + \tilde{y}_c \end{bmatrix}. \end{aligned} \quad (50)$$

For Case II, we similarly substitute the vertices  $\mathbf{v}$  from equation (45) in the centroid formula and obtain:

$$\begin{aligned} \bar{\mathbf{v}} &= \frac{1}{3} \begin{bmatrix} x_a + x_b + x_c \\ y_a + y_b + y_c \end{bmatrix} = \begin{bmatrix} (0)\beta_1 + (0)\beta_2 s \operatorname{sign}(\lambda) \frac{2}{\lambda_o} \\ (0)\beta_1 s \operatorname{sign}(\lambda) \frac{2}{\lambda_o} + (0)\beta_2 \end{bmatrix} \\ &\quad + \frac{1}{3} \begin{bmatrix} \tilde{x}_a + \tilde{x}_b + \tilde{x}_c + \operatorname{sign}(\lambda) \frac{(\tilde{y}_b - \tilde{y}_c - \tilde{y}_a + \tilde{y}_c + \tilde{y}_a - \tilde{y}_b)}{\lambda_o} \\ \tilde{y}_a + \tilde{y}_b + \tilde{y}_c - \operatorname{sign}(\lambda) \frac{(\tilde{x}_b - \tilde{x}_c - \tilde{x}_a + \tilde{x}_c + \tilde{x}_a - \tilde{x}_b)}{\lambda_o} \end{bmatrix} \\ &= \frac{1}{3} \begin{bmatrix} \tilde{x}_a + \tilde{x}_b + \tilde{x}_c \\ \tilde{y}_a + \tilde{y}_b + \tilde{y}_c \end{bmatrix}. \end{aligned} \quad (51)$$

288 *2.8. Numerical Implementation*

289 We use the theory developed in the previous sections to construct a numerically  
 290 robust procedure, given in Algorithm 1, to solve OTPPAO. **In theory, the first step**  
 291 **would be to test the input setting and then branch on Case I or Case II accordingly.**  
 292 **However, round-off errors make the test potentially unreliable. In order to deliver a**  
 293 **numerically robust solution, both cases must be attempted, and the optimal solution**  
 294 **chosen a posteriori by inspecting the cost. However, an a priori case selection method**  
 295 **for time critical but precision tolerant problems is discussed in section 2.9.** Algorithm  
 296 1 uses the input vertices  $\tilde{\mathbf{v}}$ , prescribed area  $A_o$  and orientation  $s$  as inputs. It also  
 297 uses an area error tolerance  $E$  to handle round-off in the area constraint (8). **Our**  
 298 **method could be naturally implemented with precise arithmetic, since it provides**  
 299 **exact solutions. However, in the context of its use within PBD, as presented in**  
 300 **section 3, we give it in floating-point arithmetic.** Algorithm 1 starts by generating  
 301 the solutions from Case I, then Case II, and chooses the optimal one. For Case I, we  
 302 obtain a list  $\mathbf{v}_1$  of at most 4 solutions. For Case II, we obtain a single best solution  $\mathbf{v}_2$ ,

303 the optimally rotated one, and the basis and offset to generate all solutions following  
 304 equation (49). The overall optimal solution  $\mathbf{v}_o$  is chosen amongst  $\mathbf{v}_1$  and  $\mathbf{v}_2$ . The  
 305 algorithm returns the optimal solution, along with all the solutions from Case I and  
 306 Case II. This allows the user to deal with possible ambiguities and make the final  
 307 choice depending on application specific priors and constraints.

---

**Algorithm 1** Optimal Triangle Projection with a Prescribed Area and Orientation

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\mathbf{v}_o$  - optimal triangle,  $\mathbf{v}_1$  - Case I triangle set,  $\mathbf{v}_2$  - Case II optimal triangle,  $\mathbf{v}_c, \mathbf{v}_t$  - Case II basis and translation

- 1: **function** OTTPAO( $\tilde{\mathbf{v}}, A_o, s, E = 10^{-3}$ )
  - 2:      $\mathbf{v}_1 \leftarrow \text{SOLVECASE1}(\tilde{\mathbf{v}}, A_o, s, E)$                       $\triangleright$  Compute Case I solutions
  - 3:      $(\mathbf{v}_2, \mathbf{v}_c, \mathbf{v}_t) \leftarrow \text{SOLVECASE2}(\tilde{\mathbf{v}}, A_o, s, E)$               $\triangleright$  Compute Case II solutions
  - 4:      $\mathbf{v}_o \leftarrow \text{FINDTRIANGLEOFMINIMALCOST}(\mathbf{v}_1 \cup \{\mathbf{v}_2\})$       $\triangleright$  Select the optimal solution
  - 5:     **return**  $\mathbf{v}_o, \mathbf{v}_1, \mathbf{v}_2, \mathbf{v}_c, \mathbf{v}_t$
  - 6: **end function**
- 

308     Algorithm 2 computes the possible solutions for Case I. It first computes the  
 309 coefficients  $p, q, r$  of the depressed quartic equation (lines 2, 3 and 4). Then, it  
 310 uses Ferrari's method, given by Algorithm 3, to find possible values of the Lagrange  
 311 multiplier in vector  $\boldsymbol{\lambda}$ . Some values in  $\boldsymbol{\lambda}$  may be complex because they do not  
 312 represent a solution or because of round-off error. We thus extract the real part of  $\boldsymbol{\lambda}$   
 313 (line 8). In theory, the next step would be to verify that  $\lambda \neq \lambda_o$  or  $\delta \neq 0$ , because this  
 314 would create a rank-deficiency and division by zero. However, this cannot be directly  
 315 tested because of round-off error. This is better handled by taking the pseudo-inverse  
 316  $\delta^\dagger = (3\lambda^2 - 16)^\dagger$  (line 10), recalling that  $0^\dagger = 0$ . We can then simply check that the  
 317 triangle complies with the area and orientation constraints (line 11).

318     Algorithm 4 computes all the possible solutions for Case II. It achieves this by  
 319 returning the rotational solution basis  $\mathbf{v}_c$  (line 10), particular solution  $\mathbf{v}_p$  (line 11)  
 320 and offset  $\bar{\mathbf{v}}$  (line 7). With these three components, the user can generate any  
 321 solution by choosing an angle  $\theta$  in equation (49). All solutions generated this way  
 322 are theoretically equivalent and they all fulfil the area and orientation constraints.  
 323 However, because the input vertices might not be exactly colocated numerically  
 324 (which is the theoretical prerequisite of Case II for a colocated vertices input), one of  
 325 the solutions in the basis may stand out as having a lower score than any other one.  
 326 This solution may be, when the input vertices are close to each other, the optimal

---

**Algorithm 2** Closed-form Analytic Solution to Case I of OTPPAO
 

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\nu_1$  - solution list

```

1: function SOLVECASE1( $\tilde{\mathbf{v}}, A_o, s, E$ )
2:    $p \leftarrow -\frac{16(2A_o + sA^*(\tilde{\mathbf{v}}))}{3A_o}$        $\triangleright$  Compute the coefficients of the depressed quartic
3:    $q \leftarrow \frac{32\sigma^2(\tilde{\mathbf{v}})}{3A_o}$ 
4:    $r \leftarrow \frac{256(A_o - sA^*(\tilde{\mathbf{v}}))}{9A_o}$ 
5:    $\boldsymbol{\lambda} \leftarrow \text{FERRARISOLUTION}(p, q, r)$        $\triangleright$  Solve for the four possible Lagrange
      multipliers
6:    $\nu_1 \leftarrow \emptyset$        $\triangleright$  Create an empty set of solutions
7:   for  $t \leftarrow 1, \dots, 4$  do       $\triangleright$  Generate and select the triangles
8:      $\lambda \leftarrow \text{Re}(\boldsymbol{\lambda}(t))$        $\triangleright$  Keep the real part
9:      $\delta \leftarrow 3\lambda^2 - 16$        $\triangleright$  Compute  $\delta$ 
10:     $\mathbf{v} \leftarrow \delta^\dagger \begin{bmatrix} (\lambda^2 - 16)\tilde{x}_a + \lambda^2(\tilde{x}_b + \tilde{x}_c) + 4s\lambda(\tilde{y}_b - \tilde{y}_c) \\ (\lambda^2 - 16)\tilde{y}_a + \lambda^2(\tilde{y}_b + \tilde{y}_c) + 4s\lambda(\tilde{x}_c - \tilde{x}_b) \\ (\lambda^2 - 16)\tilde{x}_b + \lambda^2(\tilde{x}_a + \tilde{x}_c) + 4s\lambda(\tilde{y}_c - \tilde{y}_a) \\ (\lambda^2 - 16)\tilde{y}_b + \lambda^2(\tilde{y}_a + \tilde{y}_c) + 4s\lambda(\tilde{x}_a - \tilde{x}_c) \\ (\lambda^2 - 16)\tilde{x}_c + \lambda^2(\tilde{x}_a + \tilde{x}_b) + 4s\lambda(\tilde{y}_a - \tilde{y}_b) \\ (\lambda^2 - 16)\tilde{y}_c + \lambda^2(\tilde{y}_a + \tilde{y}_b) + 4s\lambda(\tilde{x}_b - \tilde{x}_a) \end{bmatrix}$        $\triangleright$  Compute
      the vertices
11:    if  $|sA^*(\mathbf{v}) - A_o| \leq E$  then       $\triangleright$  Check the area constraint
12:       $\nu_1 \leftarrow \nu_1 \cup \{\mathbf{v}\}$        $\triangleright$  Add the vertices to the solution set
13:    end if
14:  end for
15:  return  $\nu_1$ 
16: end function

```

---

---

**Algorithm 3** Ferrari's Solution to the Depressed Quartic

---

**Input:**  $p, q, r$  - coefficients of depressed quartic**Output:**  $\lambda$  - set of four roots

```
1: function FERRARISOLUTION( $p, q, r$ )
2:    $Q_1 \leftarrow -\frac{p^2+12r}{36}$  ▷ Compute Cardano's formula coefficients
3:    $Q_2 \leftarrow \frac{2p^3-72rp+27q^2}{432}$ 
4:    $\alpha_o \leftarrow \sqrt[3]{Q_2 + \sqrt{Q_1^3 + Q_2^2}} + \sqrt[3]{Q_2 - \sqrt{Q_1^3 + Q_2^2}} - \frac{p}{3}$  ▷ Compute the real root
   of the resolvent cubic
5:    $\lambda \leftarrow \emptyset$  ▷ Create an empty solution set
6:   for  $k_1 \leftarrow \{1, 2\}$  do
7:     for  $k_2 \leftarrow \{1, 2\}$  do
8:        $\lambda \leftarrow \frac{(-1)^{k_1}\sqrt{2\alpha_o} + (-1)^{k_2}\sqrt{-(2p+2\alpha_o+(-1)^{k_1}\frac{2q}{\sqrt{2\alpha_o}})}}{2}$  ▷ Compute the root
9:        $\lambda \leftarrow \lambda \cup \{\lambda\}$  ▷ Add it to the solution set
10:    end for
11:  end for
12:  return  $\lambda$ 
13: end function
```

---

327 solution, even compared to Case I, owing to numerical round-off error. This solution  
328 is obtained by finding the optimal rotation for the cost function, the translation  
329 already being the optimal one, by solving:

$$\min_{R \in SO(2)} \|(\mathcal{R}\mathbf{v}_c + \mathbf{v}_t) - \tilde{\mathbf{v}}\|^2 \quad \text{with} \quad \mathcal{R} = \text{diag}(R, R, R). \quad (52)$$

330 This problem has a closed-form solution [18]. We first rearrange  $\mathbf{v}_c$  and  $\tilde{\mathbf{v}}'$  into  $2 \times 3$   
331 matrices  $V_c$  and  $\tilde{V}'$ . We then compute the cross-covariance matrix  $W = V_c \tilde{V}'^\top$  and  
332 its SVD  $W = U_1 \Sigma U_2^\top$ . The optimal orthogonal matrix, which could potentially  
333 contain a reflection in addition to the rotation, is  $U_2 U_1^\top$ . In order to preserve the  
334 triangle orientation we restrict  $R$  to be a rotation only by setting  $R = U_2 D U_1^\top$ ,  
335 where  $D = \text{diag}(1, \det(U_1 U_2^\top))$ . We finally use  $R$  to generate the optimal solution  $\mathbf{v}_2$   
336 (line 18).

337

338 *2.9. A Priori Case Selection*

339 Exact a priori case selection cannot be done to separate Cases I and II, owing to  
340 round-off errors. However, an approximate numerical approach can be implemented

---

**Algorithm 4** Closed-form Analytic Solution to Case II of OTPPAO
 

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\mathbf{v}_2$  - optimal triangle,  $\mathbf{v}_c, \mathbf{v}_t$  - triangle basis and translation

```

1: function SOLVECASE2( $\tilde{\mathbf{v}}, A_o, s, E$ )
2:   if  $|A^*(\mathbf{v})| \leq E$  then                                      $\triangleright$  Check the Input's area
3:      $k \leftarrow s \text{sign}(A^*(\mathbf{v}))$                                 $\triangleright$  Compute  $k$  for an equilateral triangle
4:   else
5:      $k \leftarrow -1$                                             $\triangleright$  Compute  $k$  for a single point
6:   end if
7:    $\bar{\mathbf{v}} \leftarrow \frac{1}{3} \begin{bmatrix} \tilde{x}_a + \tilde{x}_b + \tilde{x}_c \\ \tilde{y}_a + \tilde{y}_b + \tilde{y}_c \end{bmatrix}$         $\triangleright$  Compute the centroid of the input vertices
8:    $\tilde{\mathbf{v}}' \leftarrow \tilde{\mathbf{v}} - \bar{\mathbf{v}}$                                     $\triangleright$  Translate the input vertices
9:    $\phi \leftarrow \sqrt{\frac{\lambda_o(\text{sign}(A^*(\tilde{\mathbf{v}}))A^*(\tilde{\mathbf{v}})-4kA_o)}{12}}$     $\triangleright$  Computes the area constraint parameter
10:   $\mathbf{v}_c \leftarrow \text{Re}(\phi) \begin{bmatrix} -\frac{1}{2} & -ks\frac{2}{\lambda_o} & -\frac{1}{2} & ks\frac{2}{\lambda_o} & 1 & 0 \end{bmatrix}^\top$   $\triangleright$  Compute the solution basis
11:   $\mathbf{v}_p \leftarrow \begin{bmatrix} \frac{\tilde{x}'_a}{2} + \frac{\tilde{x}'_b}{4} + \frac{\tilde{x}'_c}{4} + \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{y}'_b - \tilde{y}'_c}{3\lambda_o} \\ \frac{\tilde{y}'_a}{2} + \frac{\tilde{y}'_b}{4} + \frac{\tilde{y}'_c}{4} - \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{x}'_b - \tilde{x}'_c}{3\lambda_o} \\ \frac{\tilde{x}'_a}{4} + \frac{\tilde{x}'_b}{2} + \frac{\tilde{x}'_c}{4} - \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{y}'_a - \tilde{y}'_c}{3\lambda_o} \\ \frac{\tilde{y}'_a}{4} + \frac{\tilde{y}'_b}{2} + \frac{\tilde{y}'_c}{4} + \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{x}'_a - \tilde{x}'_c}{3\lambda_o} \\ \frac{\tilde{x}'_a}{4} + \frac{\tilde{x}'_b}{4} + \frac{\tilde{x}'_c}{2} + \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{y}'_a - \tilde{y}'_b}{3\lambda_o} \\ \frac{\tilde{y}'_a}{4} + \frac{\tilde{y}'_b}{4} + \frac{\tilde{y}'_c}{2} - \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\tilde{x}'_a - \tilde{x}'_b}{3\lambda_o} \end{bmatrix}$   $\triangleright$  Compute the particular solution
12:   $\tilde{V}' \leftarrow$  rearrange  $\tilde{\mathbf{v}}'$  into a  $2 \times 3$  matrix
13:   $V_c \leftarrow$  rearrange  $\mathbf{v}_c$  into a  $2 \times 3$  matrix
14:   $(U_1, \Sigma, U_2) \leftarrow \text{SVD}(\tilde{V}'V_c^\top)$                   $\triangleright$  Compute the optimal rotation
15:   $D \leftarrow \text{diag}(1, \det(U_1U_2))$ 
16:   $\mathbf{v}_t \leftarrow \mathbf{v}_p + \bar{\mathbf{v}}$                                     $\triangleright$  Compute the translation vector
17:   $R \leftarrow U_2DU_1^\top$ 
18:   $\mathbf{v}_2 \leftarrow \text{diag}(R, R, R)\mathbf{v}_c + \mathbf{v}_t$                   $\triangleright$  Compute the optimal solution
19:  return  $\mathbf{v}_2, \mathbf{v}_c, \mathbf{v}_t$ 
20: end function

```

---

341 using a numerical tolerance. First, we design a measure which determines if a triangle  
 342 is equilateral or if its vertices are collocated as, according to proposition 1, this forms  
 343 a necessary condition for Case II. For that, we compute the edge lengths as  $d_{ab}$ ,  $d_{ac}$   
 344 and  $d_{bc}$  and then the maximum of the absolute edge length differences. Introducing  
 345 a numerical tolerance  $\tau_1$ , this leads to the following necessary condition for Case  
 346 II:  $\max(|d_{ab} - d_{ac}|, |d_{ab} - d_{bc}|, |d_{ac} - d_{bc}|) \leq \tau_1$ . Still following proposition 1, we  
 347 compute  $A(\tilde{\mathbf{v}})$  and  $s \text{sign}(A^*(\tilde{\mathbf{v}}))$  to determine if  $\tilde{\mathbf{v}}$  corresponds to any of the three  
 348 settings of Case II. The detailed implementation of a priori case selection is given in  
 349 Algorithm 5.

350 The numerical implementation critically depends on the two tolerance values,  $\tau_1$   
 351 and  $\tau_2$ . There is thus a risk that this test does not select the optimal solution for all  
 352 inputs, in particular inputs being very close to an equilateral triangle, but not quite  
 353 equilateral, and for which the optimal solution would be given by Case I and not  
 354 Case II. Consequently, any application relying on this method must tune  $\tau_1$  and  $\tau_2$   
 355 to maximise the instances of true positive (Case II is selected when  $\mathcal{C}(\mathbf{v}_1) \geq \mathcal{C}(\mathbf{v}_2)$ )  
 356 whilst minimising the instances of false negative selection (Case I is selected when  
 357  $\mathcal{C}(\mathbf{v}_1) \geq \mathcal{C}(\mathbf{v}_2)$ ). Furthermore, if  $\tau_1$  is very small and Case I is wrongfully selected,  
 358 then we would have  $\lambda \approx \lambda_o$  and the solution would not comply with the constraint  
 359  $sA^*(\tilde{\mathbf{v}}) - A_o \leq E$ , causing significant error.

### 360 2.10. Numerical Examples

361 We show the results of our algebraic procedure in a series of illustrative examples  
 362 presented in Tables 2 and 3. Each row represents an example with a different type  
 363 of input. The first column contains the input parameters (input vertices  $\tilde{\mathbf{v}}$  with area  
 364  $A^*(\tilde{\mathbf{v}})$ , prescribed area  $A_o$  and orientation  $s$ ). The second column shows the cost  
 365  $\mathcal{C}(\mathbf{v})$  and the generated area  $A^*(\mathbf{v})$ . For these examples, we opted to show the four  
 366 solutions from Case I and the optimally rotated solution from Case II, and highlight  
 367 the overall optimal solution. The third column shows the input triangle and the  
 368 generated solution triangles. We draw a circle and a square in two of the vertices of  
 369 the triangles to visualise the potential inversions.

370 In Table 2, the inputs are random general triangles. For each example we want to  
 371 find the optimal triangle that has the prescribed area and orientation. The first and  
 372 second examples are triangles whose orientation matches the prescribed orientation,  
 373 meaning that  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ , while the third example represents the opposite case,  
 374 meaning that  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$ . The inputs in the second and third examples are  
 375 identical, except for the prescribed orientation  $s$ . In all three examples, for Case  
 376 I, we observe that the first and second solutions respect the signed area constraint  
 377 while the third and fourth do not. This happens because the third and fourth roots



---

**Algorithm 5** A priori case selection

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $\tau_1$  - case separator tolerance 1,  $\tau_2$  - case separator tolerance 2

**Output:** selected\_case - case selection variable

```
1: function CASESELECTION( $\tilde{\mathbf{v}}, A_o, s, \tau, E$ )
2:    $d_{ab} \leftarrow |v_a - v_b|$  ▷ Edge lengths
3:    $d_{ac} \leftarrow |v_a - v_c|$ 
4:    $d_{bc} \leftarrow |v_b - v_c|$ 
5:   if  $\max(|d_{ab} - d_{ac}|, |d_{ac} - d_{bc}|, |d_{ab} - d_{bc}|) < \tau_1$  then ▷ Case Selection
6:     if  $A(\tilde{\mathbf{v}}) < \tau_2$  or  $s \text{ sign}(A^*(\tilde{\mathbf{v}})) = -1$  or  $A(\tilde{\mathbf{v}})/4 \geq A_o$  then
7:       selected_case  $\leftarrow$  2
8:     else
9:       selected_case  $\leftarrow$  1
10:    end if
11:  else
12:    selected_case  $\leftarrow$  1
13:  end if
14:  return selected_case
15: end function
```

---

378 of the depressed quartic are complex and the vertices produced by these solutions  
 379 are altered once we extract their real part in Algorithm 2. We also observe that the  
 380 third and fourth solutions of Case I are the same. The reason is that they correspond  
 381 to complex conjugate roots, and thus have the same real part. The solutions given  
 382 by Case II also respect the area constraint. In the first and second examples, the  
 383 vertices of the second solution of Case I and the solution of Case II are close to  
 384 the input vertices, resulting in lower costs, however the solution of Case II is always  
 385 an equilateral triangle. In both examples, the minimal cost is given by the second  
 386 solution of Case I and is considered optimal. In the third example, the resulting  
 387 vertices are not simple inversions of the previous solutions but new solutions that  
 388 are accommodated to the prescribed orientation. In this case an optimal solution is  
 389 also found, albeit at a higher cost.

390 In Table 3, the inputs are special configurations. The first example represents a  
 391 flat triangle with colinear input vertices. In this instance, our algorithm behaves as  
 392 expected, similarly to the examples with non-flat triangles, and returns an optimal  
 393 solution (solutions 3 and 4 of Case I return a triangle considerably larger than the  
 394 input triangle). The second example represents an example where all the input  
 395 vertices are colocated ( $\tilde{v}_a = \tilde{v}_b = \tilde{v}_c$ ). In this instance, Case I solutions are ignored  
 396 and the optimal solution is given by Case II. The solution given by Case II can be  
 397 rotated at any angle but the cost remains constant. **The third example is for an**  
 398 **equilateral triangle whose orientation is the opposite of the prescribed orientation.**  
 399 **In this example, none of the solutions given by Case I respects the area constraint**  
 400 **and  $\delta$  is very small (especially in solutions 3 and 4 of Case I where  $\delta^\dagger$  is close to zero**  
 401 **and thus returns a triangle  $10^{15}$  times larger than the input triangle). The optimal**  
 402 **solution is given by Case II and  $\mathbf{v}_c$  can be rotated at any angle, producing different**  
 403 **triangles with the cost remaining constant.** In the end, our algorithmic procedure  
 404 always computes the optimal solution for all six examples.

### 405 3. Area-based 2D Mesh Editing

406 We use our method in triangular 2D mesh editing. The implementation is similar  
 407 to PBD [6] but instead of linearising the area constraint, we perform an optimal  
 408 projection for each triangle in the mesh. The  $N_p$  mesh vertices are in  $\mathbf{P} \in \mathbb{R}^{N_p \times 2}$   
 409 and the  $N_t$  triangles in  $\mathbf{M} \in \mathbb{R}^{N_t \times 3}$  with prescribed areas  $\mathbf{A}_o \in \mathbb{R}^{N_t}$  and prescribed  
 410 orientation  $\text{sign}(A^*(\tilde{\mathbf{v}}))$ . The implementation is given in Algorithm 6.

#### 411 3.1. Shape Dataset

412 For our dataset, we used Distmesh [19] to create a set of synthetic triangular  
 413 meshes. As shown in Figure 3, the shapes ranged from simple convex shapes to

Input	Output	Generated Triangles																								
Input: Negative Oriented Triangle $s = -1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.666 & 0.666 \\ 0.666 & -0.333 \\ -1.333 & -0.333 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = -1.000$ $A_o = 0.500$	<table border="1"> <thead> <tr> <th>Cost</th> <th><math>A^*(\mathbf{v})</math></th> <th></th> </tr> </thead> <tbody> <tr> <td colspan="3">Case I</td> </tr> <tr> <td>2.820</td> <td>-0.500</td> <td style="background-color: #800000;"></td> </tr> <tr> <td><b>0.334</b></td> <td><b>-0.500</b></td> <td style="background-color: #FFA500;"></td> </tr> <tr> <td>5.065</td> <td>7.629</td> <td style="background-color: #800080;"></td> </tr> <tr> <td>5.065</td> <td>7.629</td> <td style="background-color: #008000;"></td> </tr> <tr> <td colspan="3">Case II</td> </tr> <tr> <td>0.937</td> <td>-0.500</td> <td style="background-color: #0000FF;"></td> </tr> </tbody> </table>	Cost	$A^*(\mathbf{v})$		Case I			2.820	-0.500		<b>0.334</b>	<b>-0.500</b>		5.065	7.629		5.065	7.629		Case II			0.937	-0.500		
Cost	$A^*(\mathbf{v})$																									
Case I																										
2.820	-0.500																									
<b>0.334</b>	<b>-0.500</b>																									
5.065	7.629																									
5.065	7.629																									
Case II																										
0.937	-0.500																									
Input: Positive Oriented Triangle $s = 1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.827 & -0.100 \\ 0.327 & 0.766 \\ -1.155 & -0.667 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = 1.000$ $A_o = 0.500$	<table border="1"> <thead> <tr> <th>Cost</th> <th><math>A^*(\mathbf{v})</math></th> <th></th> </tr> </thead> <tbody> <tr> <td colspan="3">Case I</td> </tr> <tr> <td>2.785</td> <td>0.500</td> <td style="background-color: #800000;"></td> </tr> <tr> <td><b>0.345</b></td> <td><b>0.500</b></td> <td style="background-color: #FFA500;"></td> </tr> <tr> <td>5.078</td> <td>-7.919</td> <td style="background-color: #800080;"></td> </tr> <tr> <td>5.078</td> <td>-7.919</td> <td style="background-color: #008000;"></td> </tr> <tr> <td colspan="3">Case II</td> </tr> <tr> <td>0.875</td> <td>0.500</td> <td style="background-color: #0000FF;"></td> </tr> </tbody> </table>	Cost	$A^*(\mathbf{v})$		Case I			2.785	0.500		<b>0.345</b>	<b>0.500</b>		5.078	-7.919		5.078	-7.919		Case II			0.875	0.500		
Cost	$A^*(\mathbf{v})$																									
Case I																										
2.785	0.500																									
<b>0.345</b>	<b>0.500</b>																									
5.078	-7.919																									
5.078	-7.919																									
Case II																										
0.875	0.500																									
Input: Positive Oriented Triangle $s = -1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.827 & -0.100 \\ 0.327 & 0.766 \\ -1.155 & -0.667 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = 1.000$ $A_o = 0.500$	<table border="1"> <thead> <tr> <th>Cost</th> <th><math>A^*(\mathbf{v})</math></th> <th></th> </tr> </thead> <tbody> <tr> <td colspan="3">Case I</td> </tr> <tr> <td>2.158</td> <td>-0.500</td> <td style="background-color: #800000;"></td> </tr> <tr> <td><b>1.041</b></td> <td><b>-0.500</b></td> <td style="background-color: #FFA500;"></td> </tr> <tr> <td>40.35</td> <td>648.58</td> <td style="background-color: #800080;"></td> </tr> <tr> <td>40.35</td> <td>648.58</td> <td style="background-color: #008000;"></td> </tr> <tr> <td colspan="3">Case II</td> </tr> <tr> <td>1.707</td> <td>-0.500</td> <td style="background-color: #0000FF;"></td> </tr> </tbody> </table>	Cost	$A^*(\mathbf{v})$		Case I			2.158	-0.500		<b>1.041</b>	<b>-0.500</b>		40.35	648.58		40.35	648.58		Case II			1.707	-0.500		
Cost	$A^*(\mathbf{v})$																									
Case I																										
2.158	-0.500																									
<b>1.041</b>	<b>-0.500</b>																									
40.35	648.58																									
40.35	648.58																									
Case II																										
1.707	-0.500																									

Table 2: Numerical examples with single triangles. The left column shows the type of input, prescribed orientation  $s$ , input vertices  $\tilde{\mathbf{v}}$ , input area  $A^*(\tilde{\mathbf{v}})$  and prescribed area  $A_o$ . The middle column shows the cost and area obtained for the triangles computed by our algebraic procedure. All four solutions from Case I and one solution from Case II are shown, assigned a colour for visual representation and the optimal solution is highlighted. The right column shows the computed triangles superimposed with the input triangle.

Input	Output		Generated Triangles
Input: Colinear Vertices $s = 1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.000 & 0.000 \\ 0.500 & 0.000 \\ 1.000 & 0.000 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = 0.000$ $A_o = 0.500$	Cost	$A^*(\mathbf{v})$	
	Case I		
	1.621	0.500	
	0.647	0.500	
	88.049	-3318	
	88.049	-3318	
	Case II		
0.762	0.500		
Input: Colocated Vertices $s = 1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.000 & 0.000 \\ 0.000 & 0.000 \\ 0.000 & 0.000 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = 0.000$ $A_o = 0.500$	Cost	$A^*(\mathbf{v})$	
	Case I		
	0.000	0.000	
	0.000	0.000	
	0.000	0.000	
	0.000	0.000	
	Case II		
1.075	0.500		
Input: Positive Oriented Equilateral Triangle $s = -1$ $\tilde{\mathbf{v}} = \begin{bmatrix} 0.000 & 0.000 \\ 1.000 & 0.000 \\ 0.500 & 0.866 \end{bmatrix}$ $A^*(\tilde{\mathbf{v}}) = 0.433$ $A_o = 0.216$	Cost	$A^*(\mathbf{v})$	
	Case I		
	0.500	0.108	
	0.500	0.108	
	$3.60 \cdot 10^{15}$	$5.62 \cdot 10^{30}$	
	$3.60 \cdot 10^{15}$	$5.62 \cdot 10^{30}$	
	Case II		
1.000	-0.216		

Table 3: Numerical examples with special triangles. The left column shows the type of input, prescribed orientation  $s$ , initial vertices  $\tilde{\mathbf{v}}$ , input area  $A^*(\tilde{\mathbf{v}})$  and prescribed area  $A_o$ . The middle column shows the cost and area obtained for the triangles computed by our algebraic procedure. All four solutions from Case I and one solution from Case II are shown and assigned a colour for visual representation. The right column shows the computed triangles superimposed with the input triangle. In the case of colocated or equilateral input vertices, the generated equilateral triangle can be rotated arbitrarily without changing the cost.

---

**Algorithm 6** Prescribed Area Preservation 2D Mesh PBD

---

**Input:**  $\mathbf{P}$  - mesh vertices,  $\mathbf{M}$  - triangle indices,  $\mathbf{A}_o$  - prescribed areas,  $T_c$  - displacement threshold,  $E$  - area error tolerance

**Output:**  $\mathbf{P}$  - edited mesh points

```
1:  $C \leftarrow \infty$ 
2: while  $C \geq T_c$  do                                ▷ Iterate until convergence
3:    $\tilde{\mathbf{P}} \leftarrow \mathbf{P}$                             ▷ Copy the mesh vertices
4:   for  $t \leftarrow 1, \dots, N_t$  do
5:      $p \leftarrow \mathbf{M}(t, :)$                             ▷ Indices of the triangle
6:      $\tilde{\mathbf{v}} \leftarrow \mathbf{P}(p, :)$                     ▷ Coordinates of the triangle
7:      $A_o \leftarrow \mathbf{A}_o(t)$                         ▷ Prescribed area of the triangle
8:      $s \leftarrow \text{sign}(A^*(\tilde{\mathbf{v}}))$                     ▷ Orientation of the triangle
9:      $\mathbf{v}_o \leftarrow \text{OTTPAO}(\tilde{\mathbf{v}}, A_o, s, E)$         ▷ Optimal projection
10:     $\mathbf{P}(p, :) \leftarrow \mathbf{v}_o$                         ▷ Update mesh points
11:   end for
12:    $C \leftarrow \frac{1}{N_t} \sum_{i=1}^{N_t} \|\tilde{\mathbf{P}}(i, :) - \mathbf{P}(i, :)\|$   ▷ Average displacement
13: end while
```

---

414 nonconvex shapes with different levels of complexity. The dataset was divided into  
415 two subsets: one subset of 8 coarse meshes composed approximately of 100 triangles  
416 and one subset of 8 fine meshes composed approximately 1000 triangles. The meshes  
417 were designed so the distances between connected vertices were approximately the  
418 same. **Our method finds the optimal triangle projection and is as such independent**  
419 **of the input triangle size. Consequently, meshes with different triangle sizes are**  
420 **seamlessly handled.** We use the areas of each of the triangles as prescribed areas  $A_o$ .

### 421 3.2. Generating Deformation Constraints

422 For our experiments we require the magnitude of the initial deformation. Since  
423 we deal with nonconvex shapes of different levels of complexity, size and orientation,  
424 we normalise this magnitude with respect to the maximum distance between two  
425 vertices in the direction of maximum variance. We treat the meshes as 2D point  
426 clouds and calculate the 95% confidence ellipse that surrounds the vertices [20]. We  
427 take the maximum distance  $D$  as twice the length of the semi-major axis of the  
428 ellipse. We divide the vertices located at the edge of the polygonal mesh in sets of  
429 connected vertices that represent a line or curve, as shown in Figure 3. Then, we  
430 apply an initial deformation by translating a given set of edge vertices in a random  
431 direction that does not cause self-collision. The magnitude of this translation is a  
432 fraction of  $D$ .

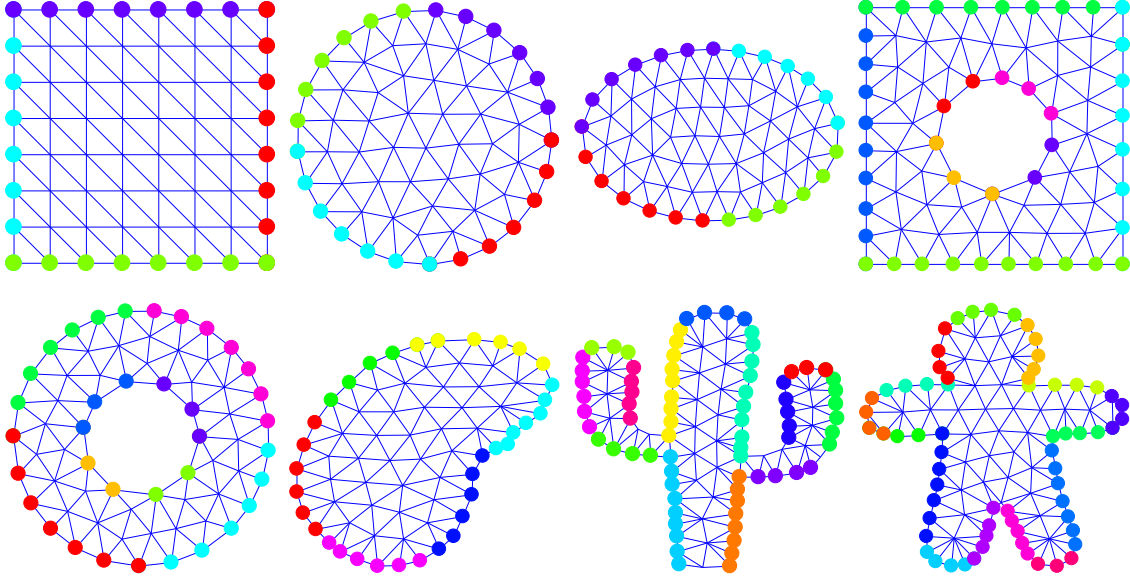


Figure 3: Shapes from the coarse subset of our synthetic polygonal mesh dataset. The vertices located at the edge are divided in sets of connected vertices represented with the same colour.

433 *3.3. Methodology*

434 We test the effectiveness of PBD-opt by applying an initial deformation to a  
 435 synthetic mesh and measuring the number iterations it takes to converge compared  
 436 to PBD-lin **on the exact same generated data**. Convergence is achieved when the  
 437 average displacement of the mesh vertices is lower than some displacement threshold  
 438  $T_c$ . We test both methods with the coarse and fine mesh datasets. For each dataset,  
 439 we perform 200 random deformations per mesh (1600 deformations in total). We  
 440 applied initial deformations to the meshes of 5%, 10% and 20% of the maximum inter-  
 441 vertex distance  $D$ . We measure the convergence speed as the number of iterations it  
 442 takes to reach three different displacement thresholds  $T_c$  at 5%, 2.5% and 1% of  $D$ .  
 443 A run stops when a method’s cost reaches the lowest threshold ( $T_c = 1\%$ ) or after it  
 444 reaches a stopping time ( $10^4$  iterations<sup>3</sup>).

445 We illustrate our methodology with an example presented in Figure 4. The input  
 446 is a circle shaped coarse mesh with an initial random deformation of 10% of maximum  
 447 inter-vertex distance  $D$ . As can be seen in Figure 4a, the initial displacement cost of

---

<sup>3</sup>The stopping time choice was arbitrary. However considering that the convergence speed of PBD-opt was lower than 1000 iterations, the stopping time is sufficiently high for our experiments.

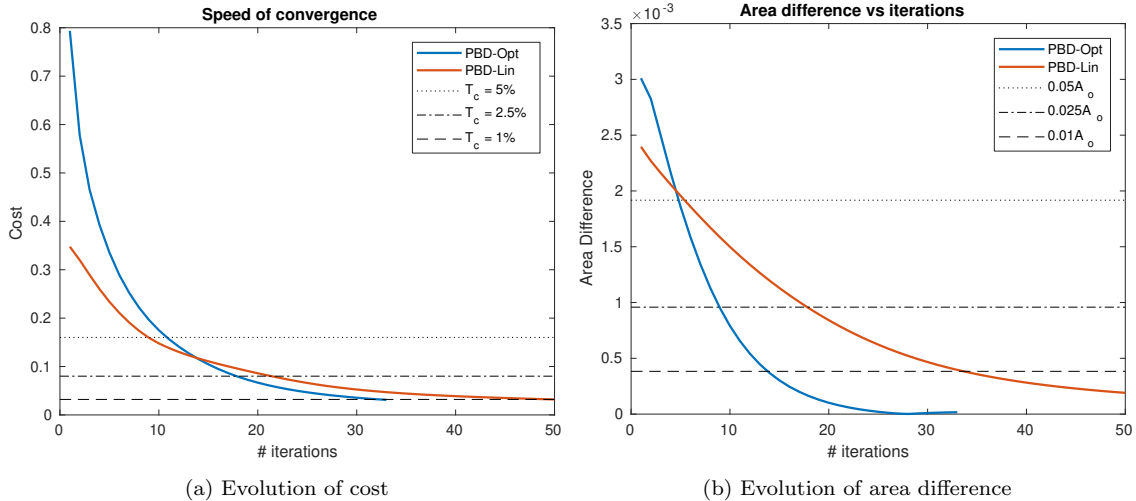


Figure 4: Displacement cost and area difference comparison of mesh-editing for both PBD-lin (red) and PBD-opt (blue) across the iterations. (a), displacement thresholds are used to quantify the evolution of convergence speed (dotted and dashed black lines). (b), area thresholds are used to quantify the evolution of the constraint convergence speed.

448 PBD-lin is lower than PBD-opt, however after some iterations the cost of PBD-opt  
 449 becomes lower while the cost of PBD-lin takes many more iterations to converge.  
 450 In Figure 4b, we show the evolution of the triangle area preservation constraint by  
 451 comparing the difference of between the mesh triangle areas and prescribed areas.  
 452 We observe that, by the time PBD-lin reaches convergence, its area difference is  
 453 larger compared to PBD-opt.

### 454 3.4. Results

455 Results can be found in Figures 5 and 6. In the left column of each Figure  
 456 we use box plots to compare the median and variability of convergence speed of  
 457 both methods according to their deformation and displacement threshold. Due to  
 458 the large number of outliers obtained, especially with PBD-lin, we decided not to  
 459 include them in the box plots but rather to represent them in stacked bar graphs in  
 460 the right column of each Figure.

461 For the coarse database we observe that the median convergence time for PBD-  
 462 opt is higher compared to PBD-lin for a threshold of 5% and relatively similar or  
 463 lower for 2.5% and 1%. However, since all the box plot pairs overlap each other, we  
 464 cannot conclude with 95% confidence that the medians differ. On the other hand, we  
 465 observe that the results of PBD-opt are more stable since they have either similar or  
 466 smaller variances compared to the results of PBD-lin. These differences are further

467 exacerbated when evaluating the fine meshes dataset where the variance of PBD-lin  
468 is many times higher compared to the variance in PBD-opt.

469 For the outlier analysis we must make a distinction between two types of outliers.  
470 The first type are Slow Convergence (SC) outliers, which surpasses the upper limit  
471 of the box plot but did not reach the stopping time. The second type are Very Slow  
472 Convergence (VSC) outliers, which reach the stopping time. For PBD-opt, at most  
473 2% of the runs were SC outliers and 0% were VSC outliers. However for PBD-lin,  
474 in the coarse meshes dataset, 10% of the runs were SC outliers. In the fine meshes  
475 dataset, around 34% of the runs were VSC outliers. This means that PBD-lin has  
476 a higher risk of getting stuck in iterations whose convergence time would be way  
477 higher compared to their median convergence speed. PBD-opt, on the other hand,  
478 provides more stable results with significantly fewer outliers.

479

### 480 *3.5. Timing Evaluation*

481 We experimentally compare the computation time of our method PBD-opt and  
482 the PBD-lin baseline. Whilst our original implementation used Matlab, the timing  
483 was done with a C implementation, required for realistic timing assessment. Then,  
484 we simulated  $10^5$  random triangles with normalised vertices  $\tilde{\mathbf{v}} \in [0, 1]$  and measured  
485 the computation time. For PBD-opt the computation time was found to be  $3.57 \cdot 10^{-6}$   
486 seconds. The computation time for one iteration of PBD-lin is  $3.79 \cdot 10^{-7}$  seconds,  
487 which is an order of magnitude faster than PBD-opt. However, the total compu-  
488 tation time for PBD-lin depends on the number of iterations required to converge,  
489 which depends on the area constraint fulfilment, hence on the constraint tolerance.  
490 Because different applications have different expectations in terms of numerical con-  
491 straint fulfilment, we ran PBD-lin with a varying constraint tolerance in the interval  
492  $[10^{-7}, 10^{-2}]$  and measured computation time for each tolerance value. The results  
493 are presented in Figure 7. As can be seen, the computation time of PBD-lin is in-  
494 versely related to the constraint tolerance, while the computation time of PBD-opt  
495 remains fixed, as expected. For low values of the constraint tolerance, which are  
496 cases where we want to be strict on the area constraint in the result, PBD-lin takes  
497 longer to converge than the optimal PBD-opt. However, when slack is introduced  
498 by increasing the constraint tolerance, which are cases where we tolerate some in-  
499 accuracy in the result of PBD-lin, PBD-lin runs faster than PBD-opt. Note that  
500 in some instances, PBD-line did not manage to fulfill the constraint (which are the  
501 VSC outliers from section 3.4) and we omitted these instances from the graph. In  
502 short, we can say that, in order to achieve a result on par with PBD-opt in terms of  
503 precision, PBD-lin takes a longer runtime in the cases where it converges properly.



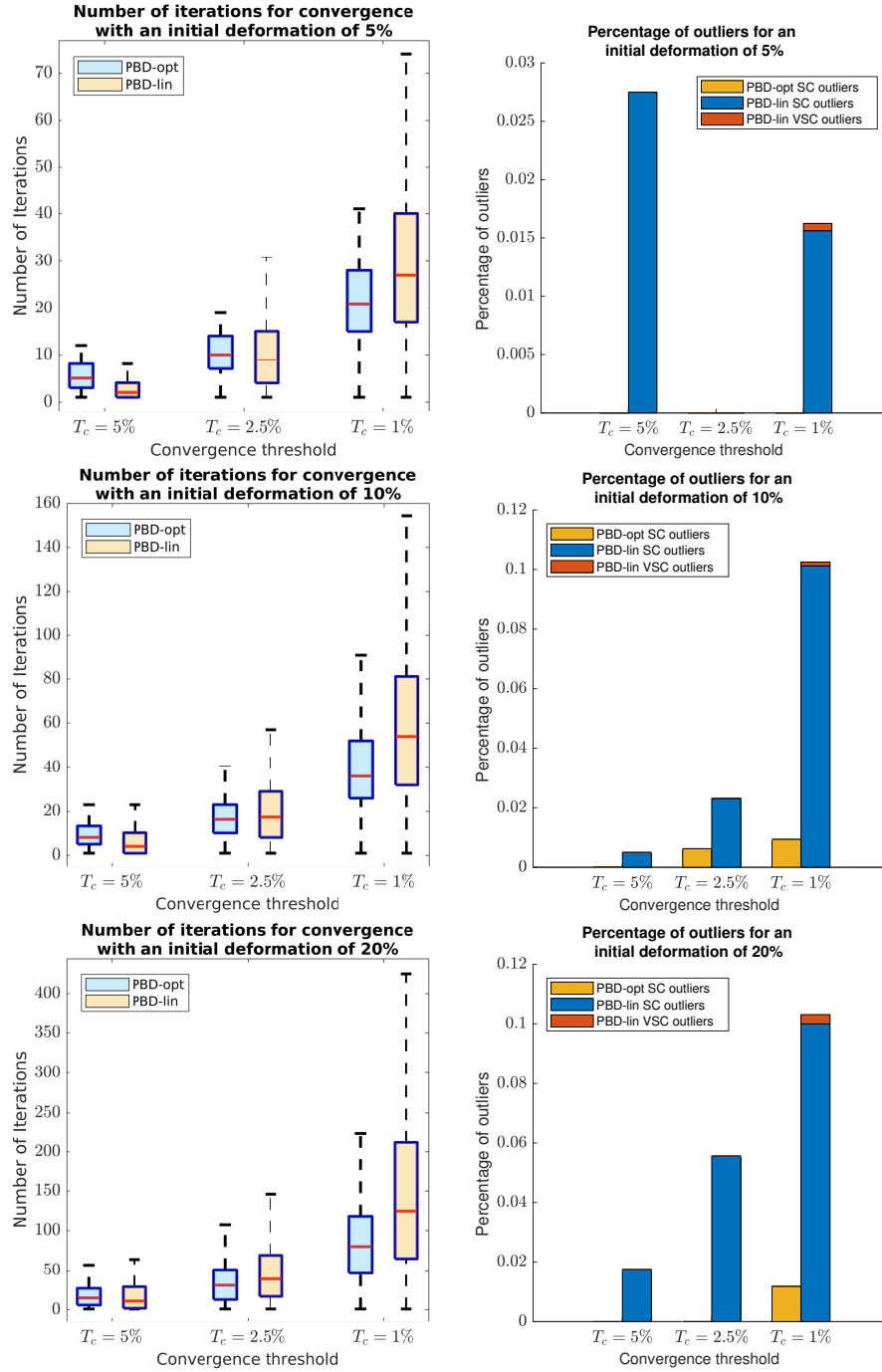


Figure 5: Convergence speed results for coarse 2D meshes. The left column shows the statistics for the number of iterations to reach conversion (omitting outliers). The right column shows the proportion of Slow Convergence (SC) and Very Slow Convergence (VSC) outliers per method.

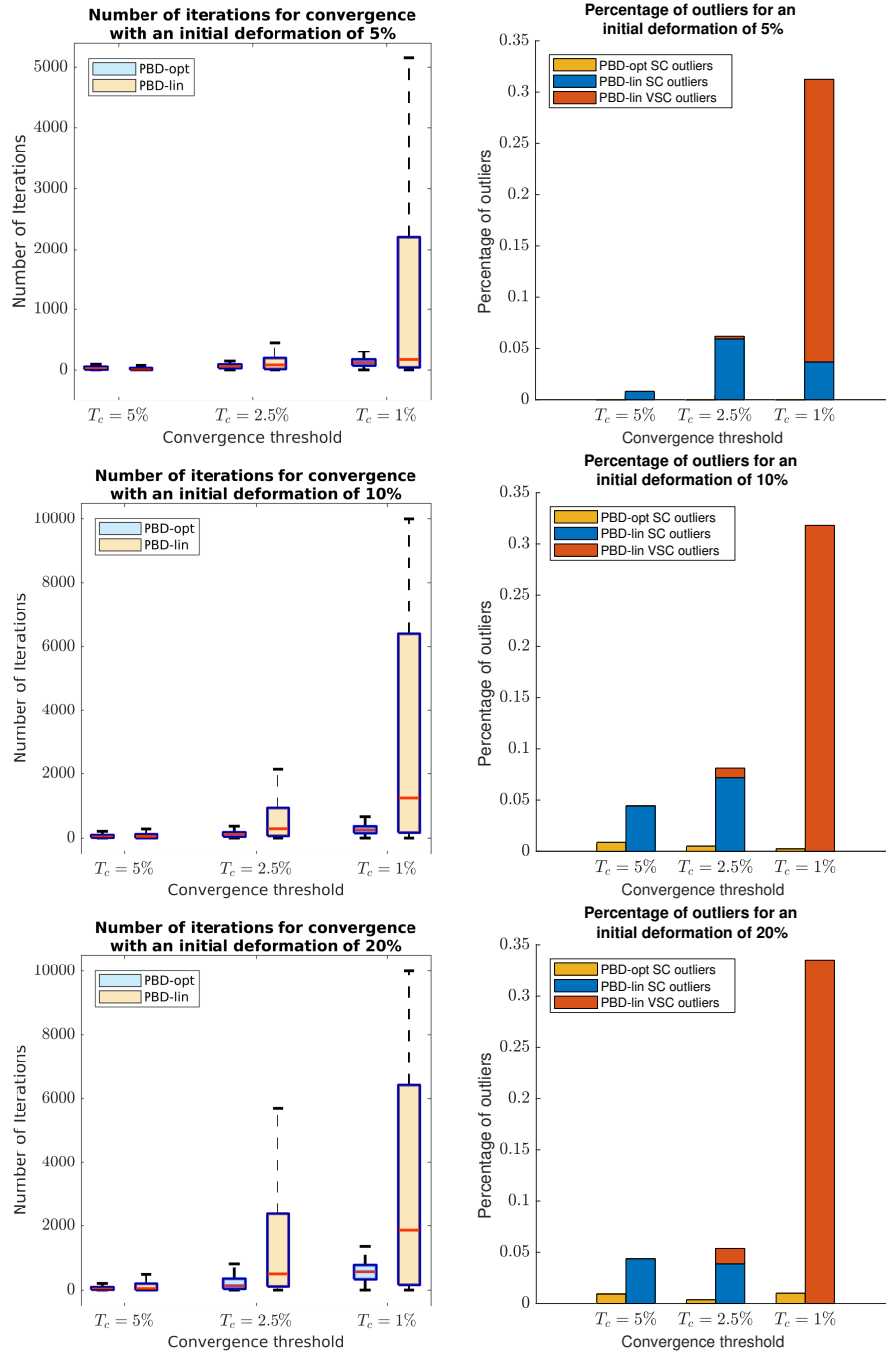


Figure 6: Convergence speed results for fine 2D meshes. The left column shows the statistics for the number of iterations to reach conversion (omitting outliers). The right column shows the proportion of Slow Convergence (SC) and Very Slow Convergence (VSC) outliers per method.

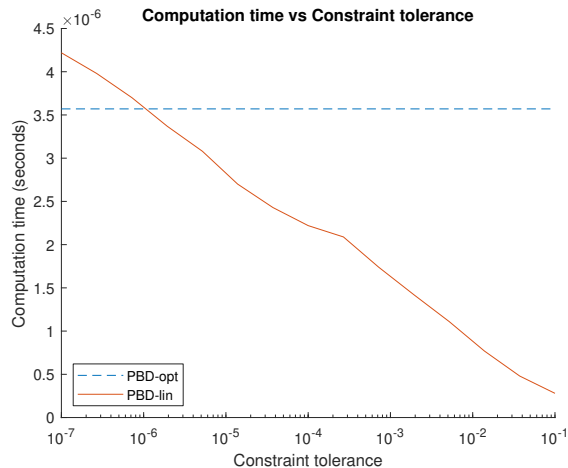


Figure 7: Computation time comparison.

504 However, whilst PBD-opt has a fixed computation budget, PBD-lin can trade-off  
 505 runtime and accuracy.

### 506 3.6. Use-cases

507 We present a series of use-cases where the triangle area and orientation constraints  
 508 are combined with additional elements such as forces and other constraints that may  
 509 be used in mesh editing. The selected elements are pin constraints (some selected  
 510 vertices are fixed during the deformation), edge length preservation and gravity. For  
 511 each use-case, we apply an initial deformation to a synthetic mesh, apply PBD-opt  
 512 and the additional forces and constraints until it converges, and compare it to PBD-  
 513 lin on the same generated data. Similarly to section 3.3, convergence is achieved  
 514 when  $T_c = 1\%$ . The notation for the different use-cases is shown in Figure 8. Note  
 515 that the initial displacement and pin constraints are applied to specific vertices,  
 516 whilst area, orientation, distance preservation constraints and gravity are applied  
 517 to all the vertices in the mesh. For each use-case we present the initial state, the  
 518 resulting deformation using both methods, the evolution of the convergence speed,  
 519 area constraint satisfaction and total displacement as shown in Figures 9, 10 and 11.  
 520 Our selected use-cases are as follows:

- 521 • **Use-case 1.** In this use-case, area and orientation must be preserved and  
 522 some pin constraints are applied. The results are shown in Figure 9. As can be  
 523 seen, PBD-opt converges faster than PBD-lin, while maintaining overall a lower  
 524 constraint satisfaction error. We also observe that under PBD-opt the mesh  
 525 vertices have an initial fast displacement until they reach a plateau, while under

526 PBD-lin they keep moving in further iterations. In this use-case, we observe  
527 that some triangles of PBD-lin reach a local minimum, as evidenced by the  
528 unexpected deformation seen in the lower right side of the mesh, which makes  
529 the overall process converge slower and have a higher constraint satisfaction  
530 error.

531 • **Use-case 2.** In this use-case, area, orientation and edge length must be pre-  
532 served. The results are shown in Figure 10. As can be seen, the overall de-  
533 formation, speed of convergence, constraint satisfaction and total displacement  
534 are almost the same for PBD-opt and PBD-lin. This is explained by the edge  
535 length preservation constraint, which has a strong impact on the deformation,  
536 making the area constraint less significant and resulting in very rigid deforma-  
537 tions.

538 • **Use-case 3.** In this use-case, area and orientation must be preserved, some pin  
539 constraints are applied and gravity is added. Gravity is implemented as a small  
540 displacement added to every non-pinned vertices of magnitude  $9.8 \cdot 10^{-4}$  at every  
541 iteration. Due to this constant displacement, it is unlikely that the simulation  
542 reaches convergence at  $T_c = 1\%$ , and we thus stopped the simulation after 300  
543 iterations. For both PBD-opt and PBD-lin, the cost and total displacement  
544 are very similar, however, PBD-opt has an overall lower constraint satisfaction  
545 error. The convergence profile for PBD-opt decreases consistently, while PBD-  
546 lin increases after iteration 200. Lastly, the simulation result is much more  
547 visually pleasing for PBD-opt than PBD-lin.

## 548 4. Conclusion

549 We have identified two problems related to finding the closest triangle to an  
550 input triangle under a prescribed area constraint (the OTPPA problem) and under a  
551 prescribed area and orientation constraints (the OTPPAO problem). We have given  
552 a detailed analysis and a closed-form solution to both of these problems for the first  
553 time. We have then developed a numerically robust algebraic implementation. We  
554 have used it within Point-Based Dynamics, resulting in a 2D triangular mesh editing  
555 procedure which has been shown to be faster and more stable than the existing  
556 method.

557 Our method forms a basis to solve further related problems in the 3D space.  
558 For a triangle in the 3D space, we can trivially apply a rigid transformation to take  
559 this triangle to one of the basis planes, calculate the optimal projection with our

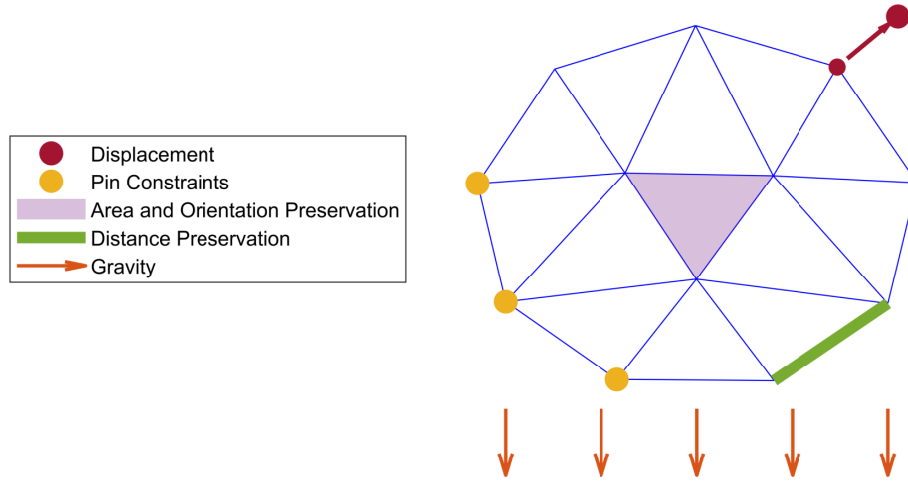


Figure 8: Use-case initial deformation and constraint notation.

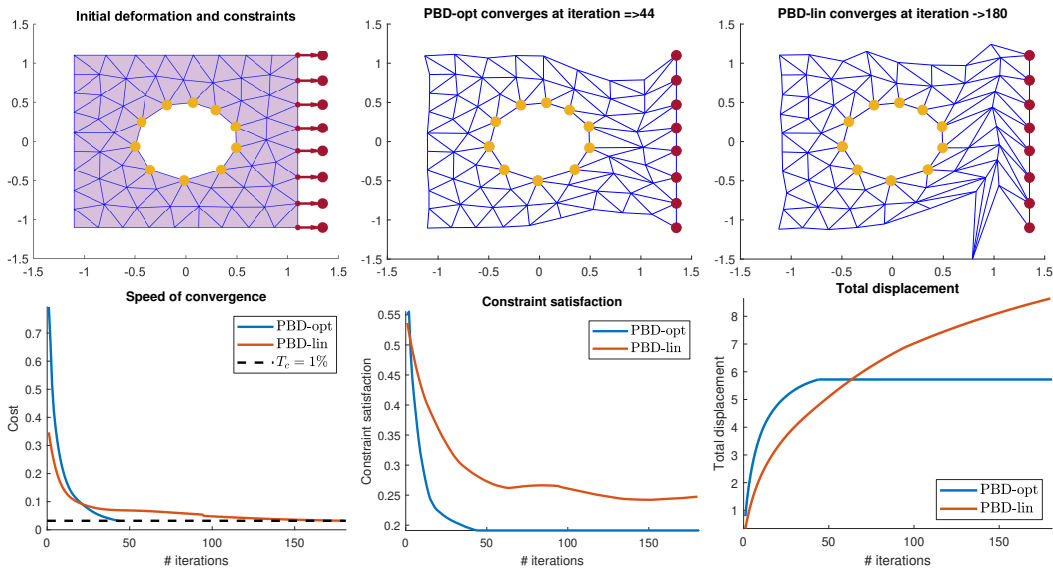


Figure 9: Results for use-case 1, with area and orientation preservation and pin constraints. The first row shows (from left to right) the initial deformation and constraints, the results from PBD-opt and PBD-lin. The second row shows the evolution of convergence speed, area constraint satisfaction and total displacement.

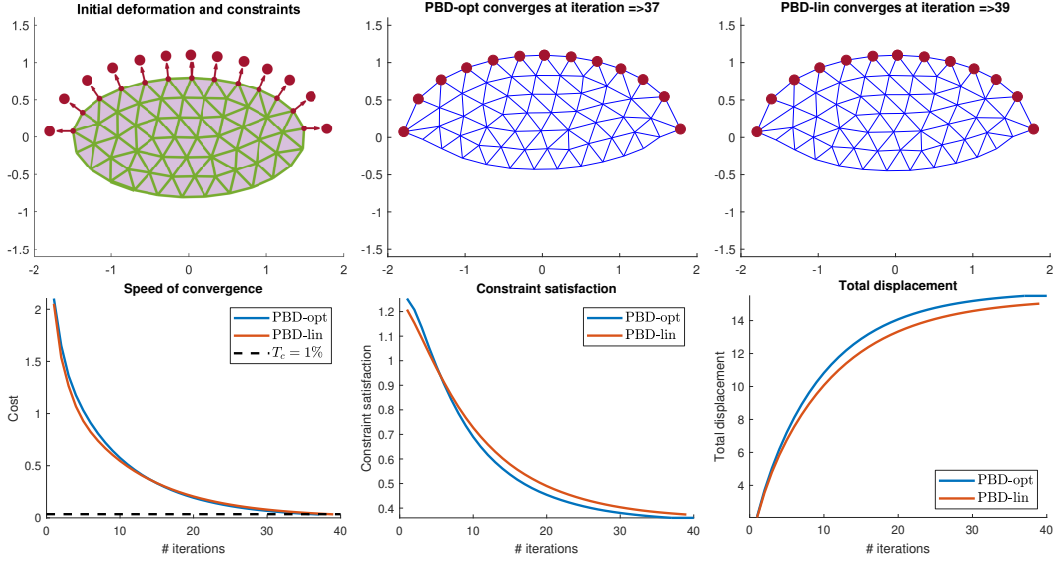


Figure 10: Results for use-case 2, with area, orientation and distance preservation. The first row shows (from left to right) the initial deformation and constraints, the results from PBD-opt and PBD-lin. The second row shows the evolution of convergence speed, area constraint satisfaction and total displacement.

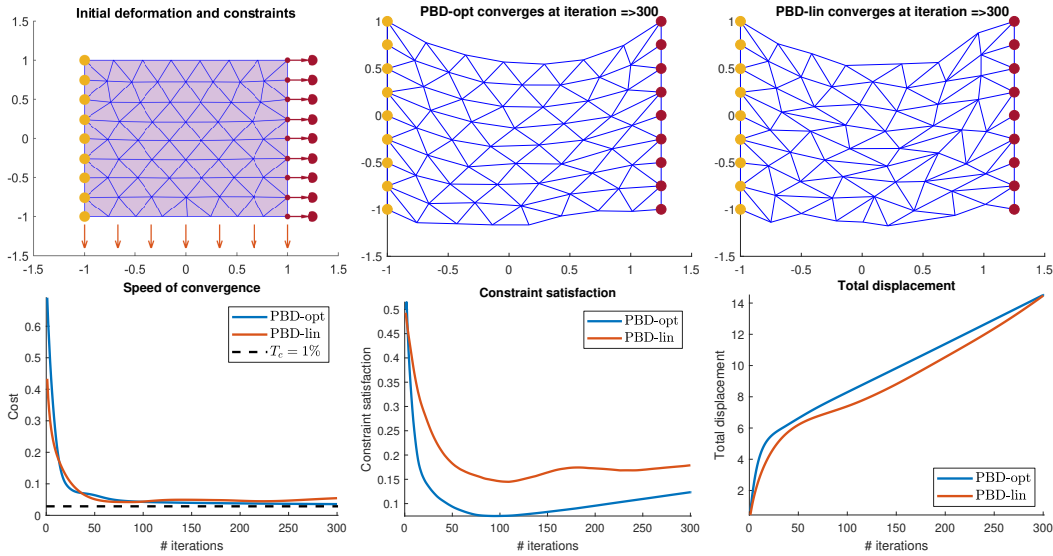


Figure 11: Results for use-case 3, with area and orientation preservation, pin constraints and gravity. The first row shows (from left to right) the initial deformation and constraints, the results from PBD-opt and PBD-lin. The second row shows the evolution of convergence speed, area constraint satisfaction and total displacement.

560 proposed 2D method and transform the vertices back to the original plane. A more  
 561 complex extension would be for a tetrahedron in the 3D space and the constraint of a  
 562 prescribed volume. This problem can be formulated similarly to OTTPAO. However,  
 563 it differs by the constraint it involves. While the area constraint in OTTPAO leads  
 564 to a quadratic equation, the volume constraint leads to a cubic equation. Therefore,  
 565 while OTTPAO provides an initial step towards solving this problem, the extension  
 566 is not a trivial one and forms the subject of future work.

## 567 Appendices

### 568 Appendix A. Proof of Lemmas

569 *Proof of Lemma 1.* We start with the forward implication:  $S_1 \Rightarrow A(\tilde{\mathbf{v}}) = 0$  and  
 570  $|\lambda| = \lambda_o$ . In  $S_1$ ,  $\tilde{\mathbf{v}}$  represents a single point. This implies  $A(\tilde{\mathbf{v}}) = 0$  and  $\sigma^2(\tilde{\mathbf{v}}) = 0$ .  
 571 Replacing these values in the depressed quartic equation (30) causes the coefficients  $p$   
 572 and  $r$  to become constants, and coefficient  $q$  to vanish (also the orientation constraint  
 573 vanishes). The depressed quartic thus transforms into a bi-quadratic:

$$\lambda^4 - \frac{32}{3}\lambda^2 + \frac{256}{9} = 0, \quad (\text{A.1})$$

574 whose solutions are:

$$|\lambda| = \lambda_o. \quad (\text{A.2})$$

575 We now turn to the reverse implication:  $S_1 \Leftarrow A(\tilde{\mathbf{v}}) = 0$  and  $|\lambda| = \lambda_o$ . We substitute  
 576  $|\lambda| = \lambda_o$  in equation (28), giving:

$$s \text{sign}(A^*(\tilde{\mathbf{v}})) \text{sign}(\lambda) \lambda_o A(\tilde{\mathbf{v}}) - \sigma^2(\tilde{\mathbf{v}}) = 0. \quad (\text{A.3})$$

577 Since  $A(\tilde{\mathbf{v}}) = 0$ , then the only solution that satisfies equation (A.3) for any given  
 578 value of  $s \text{sign}(A^*(\tilde{\mathbf{v}})) \text{sign}(\lambda)$  is with  $\sigma^2(\tilde{\mathbf{v}}) = 0$ , which implies that the input triangle  
 579 is collapsed into a single point, hence to  $S_1$ .  $\square$

580 *Proof of Lemma 2.* We start with the forward implication:  $S_2 \Rightarrow A(\tilde{\mathbf{v}}) \neq 0$  and  
 581  $\lambda = -\lambda_o$ . In  $S_2$ ,  $\tilde{\mathbf{v}}$  represents an equilateral triangle and orientation inversion. This  
 582 implies  $A(\tilde{\mathbf{v}}) \neq 0$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$ .

583 First, we show that  $\lambda$  is scale invariant; the invariance to rotation and translation  
 584 is trivial. In Case I (when  $|\lambda| \neq \lambda_o$ ),  $\lambda$  has a varying value, depending on the in-  
 585 puts. Specifically, it is resolved from the depressed quartic equation (30). Inspecting  
 586 this quartic, we trivially see that its coefficients are scale, rotation and translation  
 587 invariant. This proves that lambda, which is a root of this polynomial, is also scale,

588 translation and rotation invariant. In Case II, (when  $|\lambda| = \lambda_o$ ),  $\lambda$  has a fixed absolute  
589 value. In setting  $S_1$ , we have shown in section 2.6 that  $\text{sign}(\lambda) = -s$ , which is thus  
590 scale invariant. In settings  $S_2$  and  $S_3$ , we have shown that  $\text{sign}(\lambda) = s \text{sign}(A^*(\tilde{\mathbf{v}}))$   
591 and since  $\text{sign}(A^*(\tilde{\mathbf{v}}))$  is not affected by positive scaling, rotation or translation, then  
592  $S_2$  and  $S_3$  are also scale invariant.

Since  $\lambda$  is rotation, translation and scale invariant, we can safely perform a sim-  
ilarity transformation to  $\tilde{\mathbf{v}}$  to simplify the problem. We bring one of its vertices to  
the origin and another one to the  $x$ -axis, scaled to normalise their distance, giving  
 $\tilde{\mathbf{v}}' = [0, 0, 1, 0, 1/2, \text{sign}(A^*(\tilde{\mathbf{v}}))\sqrt{3}/2]^\top$  where  $\text{sign}(A^*(\tilde{\mathbf{v}}))$  determines the orientation  
of the triangle. We obtain  $A^*(\tilde{\mathbf{v}}') = \text{sign}(A^*(\tilde{\mathbf{v}}))\frac{\sqrt{3}}{4}$  and  $\sigma^2(\tilde{\mathbf{v}}') = 1$ , thus the coeffi-  
cients of the depressed quartic equation (30) become:

$$p = -\frac{32A_o - 4\sqrt{3}}{3A_o} \quad (\text{A.4})$$

$$q = \frac{32}{3A_o} \quad (\text{A.5})$$

$$r = \frac{256A_o + 64\sqrt{3}}{9A_o}. \quad (\text{A.6})$$

Substituting these coefficients in Cardano's formula we obtain:

$$Q_1 = -\left(\frac{32A_o + 2\sqrt{3}}{9A_o}\right)^2 \quad (\text{A.7})$$

$$Q_2 = \left(\frac{32A_o + 2\sqrt{3}}{9A_o}\right)^3, \quad (\text{A.8})$$

593 making  $\sqrt{Q_1^3 + Q_2^2} = 0$  and the real root  $\alpha_o$  of Cardano's resolvent cubic to become:

$$\alpha_o = 2\left(\frac{32A_o + 2\sqrt{3}}{9A_o}\right) + \frac{32A_o - 4\sqrt{3}}{9A_o} = \frac{32}{3}. \quad (\text{A.9})$$

594 We finally use  $\alpha_o$  to extract the roots of the depressed quartic:

$$\lambda = \frac{s_1\sqrt{2}\lambda_o + s_2\sqrt{-\lambda_o\left(\frac{s_1+1}{A_o}\right)}}{\sqrt{2}}, \quad (\text{A.10})$$

595 where  $s_1, s_2 \in \{-1, 1\}$ . We thus have the following roots:

$$\lambda \in \left\{-\lambda_o, -\lambda_o, \lambda_o - i\sqrt{\frac{\lambda_o}{A_o}}, \lambda_o + i\sqrt{\frac{\lambda_o}{A_o}}\right\}. \quad (\text{A.11})$$



596 Considering only the real roots we have  $\lambda = -\lambda_o$ .

597 We now turn to the reverse implication:  $S_2 \Leftarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = -\lambda_o$ . We  
 598 substitute  $\lambda = -\lambda_o$  in equation (28), giving:

$$-s \operatorname{sign}(A^*(\tilde{\mathbf{v}}))\lambda_o A(\tilde{\mathbf{v}}) - \sigma^2(\tilde{\mathbf{v}}) = 0. \quad (\text{A.12})$$

599 Since  $A(\tilde{\mathbf{v}}) \neq 0$  and  $\sigma^2(\tilde{\mathbf{v}}) > 0$  then equation (A.12) can only be solved when  
 600  $\operatorname{sign}(A^*(\tilde{\mathbf{v}})) = -s$ . We perform the same similarity transformation used at the begin-  
 601 ning of the proof to the unknown input triangle  $\tilde{\mathbf{v}}$  giving  $\tilde{\mathbf{v}}' = [0, 0, 1, 0, \tilde{x}'_c, \tilde{y}'_c]^\top$  where  
 602 one of the vertices remains unknown. We then have  $A^*(\tilde{\mathbf{v}}') = \frac{\tilde{y}'_c}{2}$  or  
 603  $\operatorname{sign}(A^*(\tilde{\mathbf{v}}'))A(\tilde{\mathbf{v}}') = \frac{\tilde{y}'_c}{2}$  and  $\sigma^2(\tilde{\mathbf{v}}') = \frac{2}{3}(\tilde{x}'_c{}^2 + \tilde{y}'_c{}^2 - \tilde{x}'_c + 1)$  which we substitute in  
 604 equation (A.12) and obtain:

$$\tilde{x}'_c{}^2 + \tilde{y}'_c{}^2 - \tilde{x}'_c + \sqrt{3}s\tilde{y}'_c + 1 = 0, \quad (\text{A.13})$$

605 which we rewrite as:

$$\left(\tilde{x}'_c - \frac{1}{2}\right)^2 + \left(\tilde{y}'_c + \frac{\sqrt{3}}{2}s\right)^2 = 0. \quad (\text{A.14})$$

606 This is the equation of a single point, making  $\tilde{\mathbf{v}}'$  an equilateral triangle  
 607  $\tilde{\mathbf{v}}' = [0, 0, 1, 0, 1/2, -\sqrt{3}s/2]^\top$ , where  $s$  determines the orientation of the triangle.  
 608 Since  $\tilde{\mathbf{v}}'$  was a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\tilde{\mathbf{v}}$  is also an equilateral triangle  
 609 when  $\operatorname{sign}(A^*(\tilde{\mathbf{v}})) = -s$ , which corresponds to  $S_2$ .  $\square$

*Proof of Lemma 3.* In  $S_3$ ,  $\tilde{\mathbf{v}}$  represents an equilateral triangle with  $A(\tilde{\mathbf{v}})/4 \geq A_o$  and  
 no orientation inversion. This implies  $A^*(\tilde{\mathbf{v}}) \neq 0$  and  $\operatorname{sign}(A^*(\tilde{\mathbf{v}})) = s$ . We perform  
 the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma 2, thus the coefficients of the  
 depressed quartic equation (30) become:

$$p = -\frac{32A_o + 4\sqrt{3}}{3A_o} \quad (\text{A.15})$$

$$q = \frac{32}{3A_o} \quad (\text{A.16})$$

$$r = \frac{256A_o - 64\sqrt{3}}{9A_o}. \quad (\text{A.17})$$

Substituting these coefficients in Cardano's formula we obtain:

$$Q_1 = -\left(\frac{32A_o - 4}{9A_o}\right)^2 \quad (\text{A.18})$$

$$Q_2 = -\left(\frac{32A_o - 4}{9A_o}\right)^3, \quad (\text{A.19})$$

610 making  $\sqrt{Q_1^3 + Q_2^2} = 0$ . After some factoring we obtain the real root  $\alpha_o$  of Cardano's  
 611 resolvent cubic as:

$$\alpha_o = -2\sqrt[3]{\left(\frac{32A_o - 8sA^*(\tilde{\mathbf{v}}')}{3A_o}\right)^3} + \frac{32A_o + 8sA^*(\tilde{\mathbf{v}}')}{3A_o}. \quad (\text{A.20})$$

612 Since  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  and  $A^*(\tilde{\mathbf{v}}') = \text{sign}(A^*(\tilde{\mathbf{v}}))A(\tilde{\mathbf{v}}')$ , then  $\alpha_o \in \mathbb{R}$  only when  
 613  $A(\tilde{\mathbf{v}}')/8 \geq A_o$ . Under this condition, we obtain  $\alpha_o = 32$ . Substituting in equation  
 614 (37) we obtain:

$$\lambda = \frac{s_1\sqrt{2}\lambda_o + s_2\sqrt{\lambda_o\left(\frac{s_1-1}{A_o}\right)}}{\sqrt{2}} \quad (\text{A.21})$$

615 where  $s_1, s_2 \in \{-1, 1\}$ . We thus have the following roots:

$$\lambda \in \left\{ -\lambda_o - \sqrt{\frac{\lambda_o}{A_o}}, -\lambda_o + \sqrt{\frac{\lambda_o}{A_o}}, \lambda_o, \lambda_o \right\}. \quad (\text{A.22})$$

616 Thus, we have two roots where  $|\lambda| \neq \lambda_o$  (which correspond to solutions for Case I)  
 617 and there exists at least one solution  $\lambda = \lambda_o$  for  $S_3$  (which correspond to the solution  
 618 of Case II).  $\square$

619 *Proof of Lemma 4.* We substitute  $\lambda = \lambda_o$  in equation (28), giving:

$$s \text{sign}(A^*(\tilde{\mathbf{v}}))\lambda_o A(\tilde{\mathbf{v}}) - \sigma^2(\tilde{\mathbf{v}}) = 0. \quad (\text{A.23})$$

620 Since  $A(\tilde{\mathbf{v}}) \neq 0$  then equation (A.23) can only be solved when  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ .  
 621 We perform the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma 2, substitute  
 622  $\text{sign}(A^*(\tilde{\mathbf{v}}'))A(\tilde{\mathbf{v}}')$  and  $\sigma^2(\tilde{\mathbf{v}}')$  in equation (A.23) and obtain:

$$\tilde{x}'_c{}^2 + \tilde{y}'_c{}^2 - \tilde{x}'_c + \sqrt{3}s\tilde{y}'_c + 1 = 0, \quad (\text{A.24})$$

623 which we rewrite as:

$$\left(\tilde{x}'_c - \frac{1}{2}\right)^2 + \left(\tilde{y}'_c + \frac{\sqrt{3}}{2}s\right)^2 = 0. \quad (\text{A.25})$$

624 This is the equation of a single point making  $\tilde{\mathbf{v}}'$  an equilateral triangle  
 625  $\tilde{\mathbf{v}}' = [0, 0, 1, 0, 1/2, -\sqrt{3}s/2]^\top$  where  $s$  determines the orientation of the triangle.  
 626 Since  $\tilde{\mathbf{v}}'$  was a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\tilde{\mathbf{v}}$  is also an equilateral triangle  
 627 when  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  for any value  $A(\tilde{\mathbf{v}})$  (including  $A(\tilde{\mathbf{v}})/4 \geq A_o$ ) which corre-  
 628 sponds to  $S_3$ .  $\square$

629 *Proof of Lemma 5.* We perform the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma  
630 2 and obtain  $A^*(\tilde{\mathbf{v}}') = \text{sign}(A^*(\tilde{\mathbf{v}})) \frac{\sqrt{3}}{4} = \frac{\text{sign}(A^*(\tilde{\mathbf{v}}))}{\lambda_o}$ . We relax the condition of  $S_3$  where  
631  $\frac{A(\tilde{\mathbf{v}}')}{4} \geq A_o$  by reformulating  $A_o = \frac{1}{z\lambda_o}$  where  $z$  is a scaling factor  $z > 0 \in \mathbb{R}$ . We  
632 substitute this in equation (A.22) and extract the roots :

$$\lambda = [-\lambda_o - \lambda_o\sqrt{z}, -\lambda_o + \lambda_o\sqrt{z}, \lambda_o, \lambda_o]^\top. \quad (\text{A.26})$$

633 We start with the solution given by Case I where  $|\lambda| \neq \lambda_o$ . We compact both values  
634 as  $\lambda = -\lambda_o + s_3\lambda_o\sqrt{z}$  where  $s_3 \in \{-1, 1\}$ . We substitute this in equation (22) and  
635 obtain:

$$\mathbf{v}' = \left[ \frac{\sqrt{z}-s_3}{2\sqrt{z}} \quad \frac{\sqrt{3}(\sqrt{z}-s_3)}{6\sqrt{z}} \quad \frac{\sqrt{z}+s_3}{2\sqrt{z}} \quad \frac{\sqrt{3}(\sqrt{z}-s_3)}{6\sqrt{z}} \quad \frac{1}{2} \quad \frac{\sqrt{3}(\sqrt{z}+2s_3)}{6\sqrt{z}} \right]^\top. \quad (\text{A.27})$$

636 We calculate the cost of the solution of Case I by substituting this and  $\tilde{\mathbf{v}}'$  in equation  
637 (3) and obtain:

$$\mathcal{C}_1(\mathbf{v}') = \frac{(\sqrt{z} - s_3)^2}{z} \quad (\text{A.28})$$

638 Now we turn to the solution given by Case II where  $\lambda = \lambda_o$ . We calculate the basis  
639 vector  $\mathbf{v}'_c$ , the particular solution  $\mathbf{v}'_p$  and substitute them in equation (49) and obtain:

$$\mathbf{v}' = \begin{bmatrix} \frac{1}{4} - \frac{\sqrt{3}\sqrt{z-4}}{12\sqrt{z}} \\ \frac{\sqrt{3}}{12} - \frac{\sqrt{z-4}}{4\sqrt{z}} \\ \frac{3}{4} - \frac{\sqrt{3}\sqrt{z-4}}{12\sqrt{z}} \\ \frac{\sqrt{3}}{12} + \frac{\sqrt{z-4}}{4\sqrt{z}} \\ \frac{1}{2} + \frac{\sqrt{3}\sqrt{z-4}}{6\sqrt{z}} \\ \frac{\sqrt{3}}{3} \end{bmatrix}. \quad (\text{A.29})$$

640 We calculate the cost of the solution of Case II by substituting this and  $\tilde{\mathbf{v}}'$  in equation  
641 (3) and obtain:

$$\mathcal{C}_2(\mathbf{v}') = \frac{1}{2} - \frac{1}{z} \quad (\text{A.30})$$

642 We compare the cost of both solutions  $\mathcal{C}_1(\mathbf{v}') \geq \mathcal{C}_2(\mathbf{v}')$  and obtain:

$$\frac{(\sqrt{z} - s_3)^2}{z} \geq \frac{1}{2} - \frac{1}{z} \quad (\text{A.31})$$

643 After some minor manipulations we obtain:

$$\frac{z - 4s_3\sqrt{z} + 6}{2z} \geq 0 \quad (\text{A.32})$$

644 We substitute  $\sqrt{z} = a$  where  $a > 0 \in \mathbb{R}$  and obtain the quadratic expression:

$$\frac{a^2 - 4s_3a + 6}{2a^2} \geq 0 \quad (\text{A.33})$$

645 which represents an upward opening parabola which is always positive for any value  
 646 of  $a$  and thus  $z$ . This implies that  $\mathcal{C}_1(\mathbf{v}') \geq \mathcal{C}_2(\mathbf{v}')$  for any value  $z$  including  $z > 4$ .  
 647 Since  $\tilde{\mathbf{v}}'$  was a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\mathcal{C}_1(\mathbf{v}) \geq \mathcal{C}_2(\mathbf{v})$ . This means  
 648 that in  $S_3$  the solution provided by case II has the lowest cost, thus is the optimal  
 649 solution.  $\square$

650 *Proof of Lemma 6.* We start with the forward implication:  $\mathbf{v}_p = 0 \Rightarrow \mathbf{v}'$  is an equilat-  
 651 eral triangle. From equation (43), we have that  $\mathbf{v}_p = 0$  when  $\tilde{\mathbf{v}}' = 0$  (trivial solution)  
 652 or when  $\tilde{\mathbf{v}}' = 0$  is in the kernel of  $X^\dagger$ . From the properties of the pseudo-inverse we  
 653 have that  $\ker(X^\dagger) = \ker(X^\top)$ . Since  $X$  is symmetric then  $\ker(X^\dagger) = \ker(X) = \mathbf{v}_h$ , as  
 654 given in equation (42). Recall from section 2.6 that the system  $X\mathbf{v}' = \tilde{\mathbf{v}}'$  is solvable if  
 655 and only if  $\tilde{\mathbf{v}}'$  represents an equilateral triangle of orientation  $\text{sign}(A^*(\tilde{\mathbf{v}}')) = s \text{sign}(\lambda)$   
 656 or colocated vertices. Still from section 2.6, we have that  $\mathbf{v}_h$  is an equilateral triangle  
 657 of orientation  $\text{sign}(A^*(\mathbf{v}_h)) = -s \text{sign}(\lambda)$ , which contradicts the previous statement.  
 658 This leaves us with  $\mathbf{v}_p = 0$  only when  $\tilde{\mathbf{v}}' = 0$ , which represents a set of colocated  
 659 points centred in the origin. We then calculate  $\mathbf{v}'$  from equation (45) and obtain:

$$\mathbf{v}' = \begin{bmatrix} -\frac{\beta_1}{2} + \frac{2\beta_2 \text{sign}(A^*(\tilde{\mathbf{v}}'))}{2\beta_1 \text{sign}(A^*(\tilde{\mathbf{v}}'))^{\lambda_o}} - \frac{\beta_2}{2} \\ -\frac{\beta_1}{2} - \frac{\lambda_o}{2\beta_2 \text{sign}(A^*(\tilde{\mathbf{v}}'))^2} \\ \frac{2\beta_1 \text{sign}(A^*(\tilde{\mathbf{v}}'))^{\lambda_o}}{\lambda_o} - \frac{\beta_2}{2} \\ \beta_1 \\ \beta_2 \end{bmatrix}. \quad (\text{A.34})$$

We can then easily show that the inter-vertex distances are equal, and thus that  $\mathbf{v}'$  is an equilateral triangle:

$$\begin{aligned} D_{ab} &= \sqrt{(x'_a - x'_b)^2 + (y'_a - y'_b)^2} = \sqrt{3\beta_1^2 + 3\beta_2^2} \\ D_{bc} &= \sqrt{(x'_b - x'_c)^2 + (y'_b - y'_c)^2} = \sqrt{3\beta_1^2 + 3\beta_2^2} \\ D_{ac} &= \sqrt{(x'_a - x'_c)^2 + (y'_a - y'_c)^2} = \sqrt{3\beta_1^2 + 3\beta_2^2}. \end{aligned}$$

660 We now turn to the reverse implication:  $\mathbf{v}_p = 0 \neq \mathbf{v}'$  is an equilateral triangle, for  
 661 which we simply provide a counterexample to the positive implication. We use the  
 662 equilateral triangle  $\tilde{\mathbf{v}}' = [0, 0, 1, 0, 1/2, \text{sign}(A^*(\tilde{\mathbf{v}}))\sqrt{3}/2]^\top$  where  $A(\tilde{\mathbf{v}}')/4 = A_o$  and  
 663  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ . We calculate its particular solution with equation (43) and obtain:

$$\mathbf{v}_p = \begin{bmatrix} \frac{1}{4} \\ \frac{\text{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} \\ \frac{1}{2} \\ \frac{\text{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} \\ \frac{1}{2} \\ \frac{4 \text{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} \end{bmatrix}, \quad (\text{A.35})$$

which implies that  $\mathbf{v}_p \neq 0$ . We substitute  $A(\tilde{\mathbf{v}}')/4 = A_o$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  in the constraint equation (45) and obtain  $\beta_1^2 = -\beta_2^2$ . This implies that the constraint is fulfilled when  $\beta_1 = \beta_2 = 0$  thus  $\mathbf{v}' = \mathbf{v}_p$ . We easily show that the inter-vertex distances are equal:

$$\begin{aligned} D_{ab} &= \sqrt{(x'_a - x'_b)^2 + (y'_a - y'_b)^2} = \frac{1}{2} \\ D_{bc} &= \sqrt{(x'_b - x'_c)^2 + (y'_b - y'_c)^2} = \frac{1}{2} \\ D_{ac} &= \sqrt{(x'_a - x'_c)^2 + (y'_a - y'_c)^2} = \frac{1}{2}, \end{aligned}$$

664 thus  $\mathbf{v}$  is an equilateral triangle despite  $\mathbf{v}_p \neq 0$ . □

665 *Proof of Lemma 7.* Our proof shows that the cost is invariant to the value chosen for  
 666 angle  $\theta$ . We start by translating the coordinate system to bring the input's centroid  
 667 to the origin as  $\tilde{\mathbf{v}}' = \tilde{\mathbf{v}} - \bar{\mathbf{v}}$ , which also translates the unknown vertices to  $\mathbf{v}' = \mathbf{v} - \bar{\mathbf{v}}$ .  
 668 Thus, the cost becomes the translated least-squares displacement cost:

$$\mathcal{E}'(\mathbf{v}') = \|\mathbf{v}' - \tilde{\mathbf{v}}'\|^2. \quad (\text{A.36})$$

669 We then continue with the case where  $\tilde{\mathbf{v}}$  is a single point. This implies  $A(\tilde{\mathbf{v}}) = 0$  and  
 670  $k = -1$ . The particular solution  $\mathbf{v}'_p$  can be verified to vanish, from equation (43).

671 We calculate the basis vector  $\mathbf{v}'_c$  from equation (48), leading, from equation (49), to:

$$\mathbf{v}' = \begin{bmatrix} \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \sin(\theta)}{\lambda_o} - \frac{\phi \cos(\theta)}{2} \\ -\frac{\phi \sin(\theta)}{2} - \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \cos(\theta)}{\lambda_o} \\ -\frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \sin(\theta)}{\lambda_o} - \frac{\phi \cos(\theta)}{2} \\ -\frac{\phi \sin(\theta)}{2} + \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \cos(\theta)}{\lambda_o} \\ \phi \cos(\theta) \\ \phi \sin(\theta) \end{bmatrix}. \quad (\text{A.37})$$

672 We calculate the cost by substituting  $\tilde{\mathbf{v}}$  and  $\tilde{\mathbf{v}}'$  in equation (A.36):

$$\mathcal{C}'(\mathbf{v}') = A_o \lambda_o (\sin^2(\theta) + \cos^2(\theta)), \quad (\text{A.38})$$

673 which simplifies to  $\mathcal{C}'(\mathbf{v}') = A_o \lambda_o$  and is thus independent of  $\theta$ .

674 We continue with the case where  $\tilde{\mathbf{v}}$  is an equilateral triangle. This implies that  
675  $A(\tilde{\mathbf{v}}) \neq 0$  and  $k = s \operatorname{sign}(A^*(\tilde{\mathbf{v}}))$ . We perform the same similarity transformation to  
676  $\tilde{\mathbf{v}}$  as in lemma 2 and obtain  $A^*(\tilde{\mathbf{v}}') = \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \frac{\sqrt{3}}{4} = \frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{\lambda_o}$ . We calculate the  
677 basis vector  $\mathbf{v}'_c$  from equation (48), the particular solution  $\mathbf{v}'_p$  from equation (43) and  
678 substitute them in equation (49), leading to:

$$\mathbf{v}' = \begin{bmatrix} \frac{1}{4} + \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \sin(\theta)}{\lambda_o} - \frac{\phi \cos(\theta)}{2} \\ \frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} - \frac{\phi \sin(\theta)}{2} - \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \cos(\theta)}{\lambda_o} \\ \frac{3}{4} - \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \sin(\theta)}{\lambda_o} - \frac{\phi \cos(\theta)}{2} \\ \frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} - \frac{\phi \sin(\theta)}{2} + \frac{2\phi \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \cos(\theta)}{\lambda_o} \\ \frac{1}{2} + \phi \cos(\theta) \\ \frac{4 \operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{3\lambda_o} + \phi \sin(\theta) \end{bmatrix}. \quad (\text{A.39})$$

679 We calculate the cost by substituting  $\mathbf{v}'$  and  $\tilde{\mathbf{v}}'$  in equation (A.36):

$$\mathcal{C}'(\mathbf{v}') = \frac{1}{4} + \frac{\sin^2(\theta) + \cos^2(\theta)}{4} - s \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \frac{A_o (\sin^2(\theta) + \cos^2(\theta))}{\lambda_o}, \quad (\text{A.40})$$

680 which simplifies to  $\mathcal{C}'(\mathbf{v}') = \frac{1}{2} - s \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \frac{A_o}{\lambda_o}$  and is thus independent of  $\theta$ .  $\square$

## 681 Appendix B. Optimal Triangle Projection with Prescribed Area

682 In OTPPA, only the prescribed area needs to be preserved. This means that  
683 a solution may freely choose the orientation which minimises the cost, as long as

684 the area constraint is satisfied. Consequently, we expect that OTPPA has a larger  
685 set of local extrema than OTPPAO and hence more candidate algebraic solutions.  
686 Specifically, OTPPA is stated as:

$$\min_{\mathbf{v} \in \mathbb{R}^6} \mathcal{C}(\mathbf{v}) \quad \text{s.t.} \quad f(\mathbf{v}) = 0. \quad (\text{B.1})$$

The area constraint in OTPPA is technically more complex to handle than the signed area constraint in OTPPAO, because it involves an absolute value. Fortunately, a solution may be obtained by exploiting a reformulation in terms of two rounds of OTPPAO. Similarly to OTPPAO, we start by replacing  $A(\mathbf{v})$  by  $A^*(\mathbf{v})$  in the area constraint  $f(\mathbf{v})$ , expressing  $f$  as the disjunction of two cases:

$$f(\mathbf{v}) = 0 \quad \Leftrightarrow \quad (f^+(\mathbf{v}) = 0) \vee (f^-(\mathbf{v}) = 0) \quad \text{with} \quad (\text{B.2})$$

$$f^+(\mathbf{v}) = A^*(\mathbf{v}) - A_o \quad (\text{B.3})$$

$$f^-(\mathbf{v}) = -A^*(\mathbf{v}) - A_o. \quad (\text{B.4})$$

687 We seek a solution which satisfies either  $f^+$  or  $f^-$ . This can be achieved by solving  
688 OTPPAO for  $s = 1$  and  $s = -1$ , and simply selecting the minimal cost solution a  
689 posteriori. We present the numerically robust procedure in Algorithm 7.

---

**Algorithm 7** Optimal Triangle Projection with Prescribed Area

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\mathbf{v}_o$  - optimal triangle,  $\mathbf{u}_1$  - Case I triangle set,  $\mathbf{u}_2$  - Case II triangle set,  $\mathbf{u}_c, \mathbf{u}_t$  - Case II basis and translations sets

- 1: **function** OTPPA( $\tilde{\mathbf{v}}, A_o, s, E = 10^{-3}$ )
  - 2:      $\mathbf{u}_1^- \leftarrow \text{SOLVECASE1}(\tilde{\mathbf{v}}, A_o, -1, E)$                       $\triangleright$  Compute Case I solutions
  - 3:      $\mathbf{u}_1^+ \leftarrow \text{SOLVECASE1}(\tilde{\mathbf{v}}, A_o, 1, E)$
  - 4:      $(\mathbf{v}_2^-, \mathbf{v}_c^-, \mathbf{v}_t^-) \leftarrow \text{SOLVECASE2}(\tilde{\mathbf{v}}, A_o, -1, E)$       $\triangleright$  Compute Case II solutions
  - 5:      $(\mathbf{v}_2^+, \mathbf{v}_c^+, \mathbf{v}_t^+) \leftarrow \text{SOLVECASE2}(\tilde{\mathbf{v}}, A_o, 1, E)$
  - 6:      $\mathbf{u}_1 \leftarrow \mathbf{u}_1^- \cup \mathbf{u}_1^+$                                       $\triangleright$  Add the vertices to the solution set
  - 7:      $\mathbf{u}_2 \leftarrow \{\mathbf{v}_2^-\} \cup \{\mathbf{v}_2^+\}$
  - 8:      $\mathbf{u}_c \leftarrow \{\mathbf{v}_c^-\} \cup \{\mathbf{v}_c^+\}$
  - 9:      $\mathbf{u}_t \leftarrow \{\mathbf{v}_t^-\} \cup \{\mathbf{v}_t^+\}$
  - 10:     $\mathbf{v}_o \leftarrow \text{FINDTRIANGLEOFMINIMALCOST}(\mathbf{u}_1 \cup \mathbf{u}_2)$       $\triangleright$  Select the optimal solution
  - 11:    **return**  $\mathbf{v}_o, \mathbf{u}_1, \mathbf{u}_2, \mathbf{u}_c, \mathbf{u}_t$
  - 12: **end function**
-

690 **Appendix C. Ferrari's Method for the Depressed Quartic Roots**

691 We give Ferrari's method for solving the depressed quartic equation  $\lambda^4 - p\lambda^2 +$   
 692  $q\lambda + r = 0$  ([13]).

693 **Lemma 8.** *If  $\lambda^4 + p\lambda^2 + q\lambda + r = 0$  and  $q \neq 0$  then there exists an  $\alpha_o \neq 0$  such that*

694 
$$\lambda = \frac{s_1\sqrt{2\alpha_o} + s_2\sqrt{-(2p+2\alpha_o+s_1\frac{2q}{\sqrt{2\alpha_o}})}}{2}, \text{ where } s_1, s_2 \in \{-1, 1\}.$$

695 *Proof of Lemma 8.* First, one adds and subtracts  $\frac{p^2}{4}$  to the depressed quartic equa-  
 696 tion and rewrites it as:

$$\left(\lambda^2 + \frac{p}{2}\right)^2 = -q\lambda - r + \frac{p^2}{4}. \quad (\text{C.1})$$

697 The equality to the original quartic can be verified by simple expansion. One then  
 698 introduces a variable factor  $\alpha$  into the left-hand side by adding  $2\lambda^2\alpha + p\alpha + \alpha^2$  to  
 699 both sides. Grouping the coefficients by powers of  $\lambda$  in the right-hand side gives:

$$\left(\lambda^2 + \frac{p}{2} + \alpha\right)^2 = 2\alpha\lambda^2 - q\lambda + \left(\alpha^2 + \alpha p + \frac{p^2}{4} - r\right). \quad (\text{C.2})$$

700 A quadratic expression  $ax^2 + bx + c$  is considered a perfect square when its discrim-  
 701 inant  $b^2 - 4ac = 0$  vanishes, allowing one to rewrite it as  $(\sqrt{a}x + \sqrt{c})^2$ . We use this  
 702 idea to choose a value for  $\alpha$  such that the bracketed expression in the right-hand side  
 703 of equation (C.2), which is a quadratic in  $\lambda$ , becomes a perfect square. Specifically,  
 704 vanishing the discriminant gives:

$$q^2 - 8\alpha\left(\alpha^2 + \alpha p + \frac{p^2}{4} - r\right) = 0. \quad (\text{C.3})$$

705 Upon expanding, it forms a cubic equation in  $\alpha$ , called the resolvent cubic of the  
 706 quartic equation:

$$8\alpha^3 + 8p\alpha^2 + (2p^2 - 8r)\alpha - q^2 = 0. \quad (\text{C.4})$$

707 This equation implies  $\alpha \neq 0$ . Indeed,  $\alpha = 0$  would imply  $q = 0$ , contradicting our  
 708 hypothesis  $q \neq 0$ . A real root  $\alpha_o \neq 0$  is obtained from Cardano's formula, given in  
 709 section Appendix D. Substituting in equation (C.2), we obtain:

$$\left(\lambda^2 + \frac{p}{2} + \alpha_o\right)^2 = \left(\lambda\sqrt{2\alpha_o} - \frac{q}{2\sqrt{2\alpha_o}}\right)^2. \quad (\text{C.5})$$



710 This equation is of the form  $M^2 = N^2$ , which can be rearranged as  $M^2 - N^2 = 0$  or  
 711  $(M + N)(M - N) = 0$ :

$$\left(\lambda^2 + \frac{p}{2} + \alpha_o + \lambda\sqrt{2\alpha_o} - \frac{q}{2\sqrt{2\alpha_o}}\right) \left(\lambda^2 + \frac{p}{2} + \alpha_o - \lambda\sqrt{2\alpha_o} + \frac{q}{2\sqrt{2\alpha_o}}\right) = 0. \quad (\text{C.6})$$

712 This is easily solved by applying the quadratic formula to each factor, leading to:

$$\lambda = \frac{s_1\sqrt{\alpha_o} + s_2\sqrt{-\left(p + \alpha_o + s_1\frac{q}{\sqrt{2\alpha_o}}\right)}}{\sqrt{2}}, \quad (\text{C.7})$$

713 where  $s_1, s_2 \in \{-1, 1\}$ . □

## 714 Appendix D. Cardano's Method for the Cubic Roots

We consider the cubic equation:

$$ax^3 + bx^2 + cx + d = 0 \quad \text{with} \quad a \neq 0.$$

Its solutions are:

$$\begin{aligned} x_1 &= S_1 + S_2 - \frac{b}{3a} \\ x_2 &= -\frac{S_1 + S_2}{2} - \frac{b}{3a} + \frac{i\sqrt{3}}{2}(S_1 - S_2) \\ x_3 &= -\frac{S_1 + S_2}{2} - \frac{b}{3a} - \frac{i\sqrt{3}}{2}(S_1 - S_2), \end{aligned}$$

where:

$$\begin{aligned} S_1 &= \sqrt[3]{Q_2 + \sqrt{Q_1^3 + Q_2^2}} \\ S_2 &= \sqrt[3]{Q_2 - \sqrt{Q_1^3 + Q_2^2}} \\ Q_1 &= \frac{3ac - b^2}{9a^2} \\ Q_2 &= \frac{9abc - 27a^2d - 2b^3}{54a^3}, \end{aligned}$$

and  $D = Q_1^3 + Q_2^2$  is the discriminant of the equation. For  $a, b, c, d \in \mathbb{R}$ , three cases can occur:

- (1) : if  $D > 0$ , one root is real and two are complex conjugates
- (2) : if  $D = 0$ , all roots are real, and at least two are equal
- (3) : if  $D < 0$ , all roots are real and unequal.

## 715 Appendix E. Formulation for Restricted Cases

716 Our original formulation assumes that the three triangle vertices are free to move.  
 717 However, there exist cases when one or two of the vertices are fixed. Typically, this  
 718 occurs for triangles lying on the domain boundary in mesh editing. We here adapt  
 719 the proposed optimal projection formulation to these cases.

### 720 Appendix E.1. One Fixed Vertex

#### 721 Appendix E.1.1. A Two Case Formulation

722 We assume that  $v_c$  is fixed. Thus, we have  $\mathbf{v} = [\mathbf{u}, \tilde{x}_c, \tilde{y}_c]$  and  $\tilde{\mathbf{v}} = [\tilde{\mathbf{u}}, \tilde{x}_c, \tilde{y}_c]$ , where  
 723 the moving vertices are represented by  $\mathbf{u} = [x_a, y_a, x_b, y_b] \in \mathbb{R}^4$  and the corresponding  
 724 input vertices by  $\tilde{\mathbf{u}} = [\tilde{x}_a, \tilde{y}_a, \tilde{x}_b, \tilde{y}_b] \in \mathbb{R}^4$ . We take  $\frac{\partial \mathcal{L}}{\partial \mathbf{u}} = 0$  which is formed by the  
 725 first four equalities of equation (12), which we rewrite in matrix form  $X\mathbf{u} = \mathbf{b}$  as:

$$\begin{bmatrix} 4 & 0 & 0 & s\lambda \\ 0 & 4 & -s\lambda & 0 \\ 0 & -s\lambda & 4 & 0 \\ s\lambda & 0 & 0 & 4 \end{bmatrix} \begin{bmatrix} x_a \\ y_a \\ x_b \\ y_b \end{bmatrix} = 4 \begin{bmatrix} \tilde{x}_a + s\frac{\lambda}{4}(\tilde{y}_c) \\ \tilde{y}_a - s\frac{\lambda}{4}(\tilde{x}_c) \\ \tilde{x}_b - s\frac{\lambda}{4}(\tilde{y}_c) \\ \tilde{y}_b + s\frac{\lambda}{4}(\tilde{x}_c) \end{bmatrix}. \quad (\text{E.1})$$

726 We check the invertibility of  $X$  from its determinant:

$$\det(X) = (\lambda^2 - 16)^2. \quad (\text{E.2})$$

727 We thus have:

$$\det(X) = 0 \quad \Leftrightarrow \quad |\lambda| = 4. \quad (\text{E.3})$$

728 We will see that the particular case of  $\det(X) = 0$  may occur in practice. We thus  
 729 solve system (E.1) with two cases. In Case I, which is the most general, we have  
 730  $|\lambda| \neq 4$ . In Case II, we have  $|\lambda| = 4$ . Similarly to the general case in section 2.3, the  
 731 following lemmas establish the relationship between the Lagrange multiplier and the  
 732 linear deficiency of the input vertices.

733 **Proposition 2.** *Most settings (input vertices  $\tilde{\mathbf{v}}$ , prescribed area  $A_o$  and orientation*  
 734  *$s$ ) correspond to  $F_o$  and fall in Case I. Exceptions handled with Case II are:*

- 735 •  $F_1$ :  $\tilde{\mathbf{v}}$  is a single point
- 736 •  $F_2$ :  $\tilde{\mathbf{v}}$  is a right isosceles triangle and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$
- 737 •  $F_3$ :  $\tilde{\mathbf{v}}$  is a right isosceles triangle,  $A(\tilde{\mathbf{v}})/4 \geq A_o$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$

738 The proof of Proposition 2 is based on the following five lemmas.

739 **Lemma 9.**  $F_1 \iff A(\tilde{\mathbf{v}}) = 0$  and  $|\lambda| = 4$ .

740 **Lemma 10.**  $F_2 \iff A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = -4$ .

741 **Lemma 11.**  $F_3 \implies A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda \in \left\{ 4, -4 + 2\sqrt{\frac{2}{A_o}}, -4 - 2\sqrt{\frac{2}{A_o}} \right\}$ .

742 **Lemma 12.**  $F_3 \Leftarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = 4$ .

743 **Lemma 13.**  $\lambda = 4$  leads to the optimal solution for  $F_3$ .

744 The proofs of these lemmas are given in Appendix E.1.4. It is important to clarify  
 745 for Proposition 2, that in the right isosceles triangle, the fixed vertex corresponds to  
 746 the one opposite to the triangle's hypotenuse.

747 *Proof of Proposition 2.* We recall that Case I occurs for  $|\lambda| \neq 4$  and Case II for  
 748  $|\lambda| = 4$ . Lemmas 7, 8 and 10 show that  $F_1$ ,  $F_2$  and  $F_3$  are the only possible settings  
 749 corresponding to  $|\lambda| = \lambda_o$ , hence possibly to Case II. This proves that Case I is the  
 750 general case. Lemmas 7 and 8 then trivially prove that  $F_1$  and  $F_2$  are handled by  
 751 Case II. Finally, lemmas 9 and 11 prove that  $F_3$  is also handled by Case II.  $\square$

### 752 Appendix E.1.2. Case I

753 This case occurs for  $|\lambda| \neq 4$ . In other words, this is the case where at least one of  
 754 the initial vertices in  $\tilde{\mathbf{u}}$  is different from the other two and where the rank of the input  
 755 matrix  $M$  in equation (18) is  $\text{rank}(M) > 1$ . Similarly to Case I for three moving  
 756 vertices, the problem is reformulated as a depressed quartic polynomial and the  
 757 roots of this polynomial are found using Ferrari's method. We start by multiplying  
 758 equation (E.1) by the adjugate  $X^*$  of  $X$  and obtain:

$$\det(X)\mathbf{u} = X^*\mathbf{b}, \tag{E.4}$$

759 where the adjugate is:

$$X^* = \delta Y = \lambda^2 - 16 \begin{bmatrix} -4 & 0 & 0 & s\lambda \\ 0 & -4 & -s\lambda & 0 \\ 0 & -s\lambda & -4 & 0 \\ s\lambda & 0 & 0 & -4 \end{bmatrix}, \tag{E.5}$$

760 with  $\delta = \lambda^2 - 16$  and  $Y \in \mathbb{R}^{4 \times 4}$ . We note that  $\det(X) = \delta^2$ . We substitute this and  
 761 equation (E.5) in equation (E.4) and obtain:

$$\delta \mathbf{u} = Y \mathbf{b}. \quad (\text{E.6})$$

762 We observe that the signed area  $A^*(\delta \mathbf{v}) = \delta^2 A^*(\mathbf{v})$ . Also that  $\delta \mathbf{v} = [\mathbf{u}, \delta \tilde{x}_c, \delta \tilde{y}_c]$ .  
 763 Thus, we calculate the signed area of  $\delta \mathbf{v}$  and obtain after some minor expanding:

$$\delta^2 A^*(\mathbf{v}) = a_2 \lambda^2 + a_1 \lambda + a_o, \quad (\text{E.7})$$

where:

$$a_0 = 128((\tilde{x}_a - \tilde{x}_c)(\tilde{y}_b - \tilde{y}_a) - (\tilde{x}_a - \tilde{x}_b)(\tilde{y}_c - \tilde{y}_a)) \quad (\text{E.8})$$

$$a_1 = -32(\tilde{x}_a^2 + \tilde{x}_b^2 + 2\tilde{x}_c^2 + \tilde{y}_a^2 + \tilde{y}_b^2 + 2\tilde{y}_c^2) \quad (\text{E.9})$$

$$+ 64(\tilde{x}_a \tilde{x}_b + \tilde{x}_a \tilde{x}_c + \tilde{x}_b \tilde{x}_c + \tilde{y}_a \tilde{y}_b + \tilde{y}_a \tilde{y}_c + \tilde{y}_b \tilde{y}_c) \quad (\text{E.10})$$

$$a_2 = 8((\tilde{x}_a - \tilde{x}_c)(\tilde{y}_b - \tilde{y}_a) - (\tilde{x}_a - \tilde{x}_b)(\tilde{y}_c - \tilde{y}_a)). \quad (\text{E.11})$$

764 We rewrite these coefficients more compactly. Concretely,  $a_0$  and  $a_2$  contain the  
 765 signed area  $A^*(\tilde{\mathbf{v}})$  of the input vertices and  $a_1$  contains the sum of the squared  
 766 distances  $D_o$  of the two moving vertices to the fixed vertex:

$$D_o(\tilde{\mathbf{v}}) = (\tilde{x}_a - \tilde{x}_c)^2 + (\tilde{y}_a - \tilde{y}_c)^2 + (\tilde{x}_b - \tilde{x}_c)^2 + (\tilde{y}_b - \tilde{y}_c)^2. \quad (\text{E.12})$$

767 We substitute equation (E.7) in the signed area constraint (8) multiplied by  $\delta^2$  and  
 768 obtain:

$$sA^*(\mathbf{v}) - \delta^2 A_o = 0. \quad (\text{E.13})$$

769 This way, the signed area only depends on the known initial vertices  $\tilde{\mathbf{v}}$  and prescribed  
 770 sign  $s$ . Because the signed area is quadratic in the vertices, and the vertices are  
 771 quadratic rational in  $\lambda$ , the resulting equation is a quartic in  $\lambda$ :

$$A_o \lambda^4 - 16(2A_o + sA^*(\tilde{\mathbf{v}}))\lambda^2 + 32D_o(\tilde{\mathbf{v}})\lambda + 256(A_o - sA^*(\tilde{\mathbf{v}})) = 0. \quad (\text{E.14})$$

772 This is a depressed quartic because it does not have a cubic term. We can thus  
 773 rewrite it to the standard form by simply dividing by  $A_o$ , giving:

$$\lambda^4 + p\lambda^2 + q\lambda + r = 0, \quad (\text{E.15})$$

with:

$$p = -\frac{16(2A_o + sA^*(\tilde{\mathbf{v}}))}{A_o} \quad (\text{E.16})$$

$$q = \frac{32D_o(\tilde{\mathbf{v}})}{A_o} \quad (\text{E.17})$$

$$r = \frac{256(A_o - sA^*(\tilde{\mathbf{v}}))}{A_o}. \quad (\text{E.18})$$

774 This can be solved using Ferrari's method. We observe that when  $\text{rank}(M) = 2$ ,  
 775 implying  $A^*(\tilde{\mathbf{v}}) = 0$ , coefficients  $p$  and  $r$  become constants. However, the equation  
 776 remains a general depressed quartic which can be solved with our procedure.

777 *Appendix E.1.3. Case II*

778 Case II occurs for  $|\lambda| = 4$ . From Proposition 2, this means that the initial  
 779 vertices  $\tilde{\mathbf{v}}$  are collocated as  $\tilde{v}_a = \tilde{v}_b = \tilde{v}_c$  or represent right isosceles triangles under  
 780 conditions from Proposition 2. We show that the problem is represented by translated  
 781 homogeneous and linearly dependent equations. We find their null space, particular  
 782 solution and then a subset constrained by the prescribed area.

783 We first translate the coordinate system to bring the fixed vertex to the origin  
 784  $\tilde{\mathbf{u}}' = \tilde{\mathbf{u}} - v_c$  translating the unknown vertices to  $\mathbf{u}' = \mathbf{u} - v_c$ . Substituting  $|\lambda| = 4$  in  
 785 matrix  $X$ , we obtain:

$$4 \begin{bmatrix} 1 & 0 & 0 & s \text{sign}(\lambda) \\ 0 & 1 & -s \text{sign}(\lambda) & 0 \\ 0 & -s \text{sign}(\lambda) & 1 & 0 \\ s \text{sign}(\lambda) & 0 & 0 & 1 \end{bmatrix}. \quad (\text{E.19})$$

786 This matrix has a rank of two and is thus non-invertible. Thus,  $X\mathbf{u}' = \tilde{\mathbf{u}}'$  is solvable  
 787 if and only if  $\tilde{\mathbf{u}}'$  lies in the column space  $C(X)$ . The column space can be calculated  
 788 by factoring  $X$  into its singular value decomposition (SVD)  $X = W\Sigma W^\top$  and taking  
 789 the first  $\text{rank}(X)$  columns of the unitary matrix  $W$ . For each value of  $s \text{sign}(\lambda)$ , we  
 790 have column spaces expressed as two-dimensional linear subspaces  $\{\gamma_1 \mathbf{w}_1^- + \gamma_2 \mathbf{w}_2^-\}$   
 791 and  $\{\gamma_1 \mathbf{w}_1^+ + \gamma_2 \mathbf{w}_2^+\}$  where  $\gamma_1, \gamma_2 \in \mathbb{R}$  and with bases  $\mathbf{w}_1^-, \mathbf{w}_2^- \in \mathbb{R}^4$  and  $\mathbf{w}_1^+, \mathbf{w}_2^+ \in \mathbb{R}^4$   
 792 such that:

$$[\mathbf{w}_1^- \quad \mathbf{w}_2^-] = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 \\ \frac{1}{\sqrt{2}} & 0 \\ 0 & -\frac{1}{\sqrt{2}} \end{bmatrix}, \quad (\text{E.20})$$

793 and:

$$[\mathbf{w}_1^+ \quad \mathbf{w}_2^+] = \begin{bmatrix} 0 & \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & 0 \\ -\frac{1}{\sqrt{2}} & 0 \\ 0 & \frac{1}{\sqrt{2}} \end{bmatrix}, \quad (\text{E.21})$$

794 We have that when we add a vertex located in the origin to  $\mathbf{w}_1^-, \mathbf{w}_2^-, \mathbf{w}_1^+$  and  $\mathbf{w}_2^+$ , they  
 795 represent right isosceles triangles of the same area of  $\frac{1}{4}$  with orientation  $s \text{sign}(\lambda)$ .  
 796 Similarly, adding a vertex located in the origin to the linear combinations  $\gamma_1 \mathbf{w}_1^- +$

797  $\gamma_2 \mathbf{w}_2^-$  and  $\gamma_1 \mathbf{w}_1^+ + \gamma_2 \mathbf{w}_2^+$  represent right isosceles triangles of any area and opposite  
 798 orientations (or colocated points if  $\gamma_1 = \gamma_2 = 0$ ). This shows that the system is  
 799 solvable if and only if  $\mathbf{v}' = [\mathbf{u}', 0, 0] \in \mathbb{R}^6$  represents a right isosceles triangle of  
 800 orientation  $\text{sign}(A^*(\tilde{\mathbf{v}}')) = s \text{sign}(\lambda)$  or colocated vertices.

801 The system  $X\mathbf{u}' = \tilde{\mathbf{u}}'$  is solved by first finding the solutions of the homogeneous  
 802 system  $X\mathbf{u}_h = 0$  and translating them by a particular solution  $\mathbf{u}_p$ , obtaining  $\mathbf{u}' =$   
 803  $\mathbf{u}_h + \mathbf{u}_p$ . The homogeneous system has an infinite number of solutions which come  
 804 from the null space of  $X$ . This can be represented as a two-dimensional linear  
 805 subspace  $\mathbf{u}_h = \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2$  where the coefficients  $\beta_1, \beta_2 \in \mathbb{R}$  and with bases  $\mathbf{u}_1, \mathbf{u}_2 \in$   
 806  $\mathbb{R}^4$  such that:

$$[\mathbf{u}_1 \quad \mathbf{u}_2] = \begin{bmatrix} 0 & -s \text{sign}(\lambda) \\ s \text{sign}(\lambda) & 0 \\ 1 & 0 \\ 0 & 1 \end{bmatrix}. \quad (\text{E.22})$$

807 We have that  $\mathbf{u}_1$  and  $\mathbf{u}_2$  represent two pairs of vertices which form right isosceles  
 808 triangles when adding one third vertex in the origin of the same area  $\frac{1}{4}$ . The linear  
 809 combination of  $\mathbf{u}_h = \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2$  generate right isosceles triangles of any area when  
 810 adding one third vertex in the origin of orientation  $-s \text{sign}(\lambda)$ . We then calculate  
 811 the particular solution  $\mathbf{u}_p$  using the pseudo-inverse as:

$$\mathbf{u}_p = X^\dagger \tilde{\mathbf{u}}' = \frac{1}{4} \begin{bmatrix} \tilde{x}'_a + \text{sign}(A^*(\tilde{\mathbf{v}}')) \tilde{y}'_b \\ \tilde{y}'_a - \text{sign}(A^*(\tilde{\mathbf{v}}')) \tilde{x}'_b \\ \tilde{x}'_b - \text{sign}(A^*(\tilde{\mathbf{v}}')) \tilde{y}'_a \\ \tilde{y}'_b + \text{sign}(A^*(\tilde{\mathbf{v}}')) \tilde{x}'_a \end{bmatrix} \quad (\text{E.23})$$

812 We then translate the null space with the particular solution and obtain  $\mathbf{u}' = \beta_1 \mathbf{u}_1 +$   
 813  $\beta_2 \mathbf{u}_2 + \mathbf{u}_p$ . This linear combination represents pairs of vectors. **A noticeable property**  
 814 **is stated in the following lemma, whose proof is given in Appendix E.1.4.**

815 **Lemma 14.**  $\mathbf{u}_p = 0 \Rightarrow \mathbf{v}' = [\mathbf{u}', 0, 0]$  *is a right isosceles triangle. The converse is*  
 816 *not true.*

817 The next step is to constrain these to the prescribed area and orientation. We have  
 818  $\mathbf{v}' = [\mathbf{u}', 0, 0] \in \mathbb{R}^6$ . After some minor algebraic manipulations, we obtain the signed  
 819 area as:

$$A^*(\mathbf{v}') = (1 + s \text{sign}(A^*(\tilde{\mathbf{v}}')) \text{sign}(\lambda)) \frac{A^*(\tilde{\mathbf{v}}')}{4} - s \text{sign}(\lambda) \frac{\beta_1^2 + \beta_2^2}{2}. \quad (\text{E.24})$$

820 When  $A^*(\tilde{\mathbf{v}}') \neq 0$  we have  $s \text{sign}(\lambda) = \text{sign}(A^*(\tilde{\mathbf{v}}'))$  thus  $\text{sign}(\lambda) = s \text{sign}(A^*(\tilde{\mathbf{v}}'))$ .  
 821 However, when  $A^*(\tilde{\mathbf{v}}') = 0$  we have  $\text{sign}(A^*(\mathbf{v}')) = -s \text{sign}(\lambda)$ , which implies that

822  $\text{sign}(\lambda) = -s$ . Using the orientation constraint (7), we can express  $\mathbf{u}'$  as:

$$\begin{aligned} \mathbf{u}' &= \beta_1 \mathbf{u}_1 + \beta_2 \mathbf{u}_2 + \mathbf{u}_p \\ \text{s.t.} \quad & (s + \text{sign}(\lambda) \text{sign}(A^*(\tilde{\mathbf{v}}'))) \frac{A^*(\tilde{\mathbf{v}}')}{4} - \text{sign}(\lambda) \frac{(\beta_1^2 + \beta_2^2)}{2} - A_o = 0 \end{aligned} \quad (\text{E.25})$$

823 Because of the area and orientation constraints, and because  $\mathbf{u}_1$  and  $\mathbf{u}_2$  are rotated  
824 copies of each other, the family defined by equation (E.22) can be generated by  
825 simply rotating  $\mathbf{u}_1$  by

$$\rho = \sqrt{\beta_1^2 + \beta_2^2} = \sqrt{\frac{\text{sign}(A^*(\tilde{\mathbf{v}}'))A^*(\tilde{\mathbf{v}}') - 4kA_o}{2}} \quad (\text{E.26})$$

826 where  $k$  depends on the type of input:

$$k = \begin{cases} s \text{sign}(A^*(\tilde{\mathbf{v}}')) & \text{if } A^*(\tilde{\mathbf{v}}) \neq 0 \\ -1 & \text{if } A^*(\tilde{\mathbf{v}}) = 0 \end{cases} \quad (\text{E.27})$$

827 so that the area constraint is met and then rotate  $\mathbf{u}_1$  by some arbitrary angle  $\theta$ . We  
828 note that when  $k = s \text{sign}(A^*(\tilde{\mathbf{v}}')) = 1$ , then  $\rho \in \mathbb{R}$  as long as  $A(\tilde{\mathbf{v}})/4 \geq A_o$  (which  
829 corresponds to setting  $F_3$ ). We define a new basis vector  $\mathbf{u}_c$  as:

$$\mathbf{u}_c = \rho \begin{bmatrix} 0 & ks & 1 & 0 \end{bmatrix}. \quad (\text{E.28})$$

830 Finally the triangle vertices are translated to the original coordinates by adding the  
831 fixed vertex  $v_c$ , we obtain:

$$\mathbf{v} = [\mathcal{R}(\theta)\mathbf{u}_c + \mathbf{u}_p \quad 0 \quad 0]^\top + [v_c \quad v_c \quad v_c]^\top \quad (\text{E.29})$$

832 where  $\mathcal{R}(\theta)$  is a block diagonal matrix replicating the 2D rotation matrix  $R(\theta)$  two  
833 times as  $\mathcal{R} = \text{diag}(R(\theta), R(\theta))$ . **Equation (E.29) generates an infinite number of solu-**  
834 **tions with the same cost, as shown by the following lemma, proved in Appendix E.1.4.**

835 **Lemma 15.** *All solutions generated by equation (E.29) have the same cost.*

836 *Appendix E.1.4. Proof of Lemmas*

837 *Proof of Lemma 9.* We start with the forward implication:  $F_1 \Rightarrow A(\tilde{\mathbf{v}}) = 0$  and  
838  $|\lambda| = 4$ . In  $F_1$ ,  $\tilde{\mathbf{v}}$  represents a single point. This implies  $A(\tilde{\mathbf{v}}) = 0$  and  $D_o(\tilde{\mathbf{v}}) = 0$ .  
839 Replacing these values in the depressed quartic equation (30) causes the coefficients  $p$

840 and  $r$  to become constants, and coefficient  $q$  to vanish (also the orientation constraint  
841 vanishes). The depressed quartic thus transforms into a bi-quadratic:

$$\lambda^4 - 32\lambda^2 + 256 = 0, \quad (\text{E.30})$$

842 whose solutions are:

$$|\lambda| = 4. \quad (\text{E.31})$$

843 We now turn to the reverse implication:  $F_1 \Leftarrow A(\tilde{\mathbf{v}}) = 0$  and  $|\lambda| = 4$ . We substitute  
844  $|\lambda| = 4$  in equation (E.13), giving:

$$4s \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \operatorname{sign}(\lambda) |A^*(\tilde{\mathbf{v}})| - D_o(\tilde{\mathbf{v}}) = 0. \quad (\text{E.32})$$

845 Since  $A(\tilde{\mathbf{v}}) = 0$ , then the only solution that satisfies equation (E.32) for any given  
846 value of  $s \operatorname{sign}(A^*(\tilde{\mathbf{v}})) \operatorname{sign}(\lambda)$  is with  $D_o(\tilde{\mathbf{v}}) = 0$ , which implies that the input triangle  
847 is collapsed into a single point, hence to  $F_1$ .  $\square$

*Proof of Lemma 10.* We start with the forward implication:  $F_2 \Rightarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = -4$ . In  $F_2$ ,  $\tilde{\mathbf{v}}$  represents a right isosceles triangle and orientation inversion. This implies  $A(\tilde{\mathbf{v}}) \neq 0$  and  $\operatorname{sign}(A^*(\tilde{\mathbf{v}})) = -s$ . We perform a similarity transformation to  $\tilde{\mathbf{v}}$  to bring the fixed vertex to the origin and another to the  $x$ -axis, scaled to normalise their distance, giving  $\tilde{\mathbf{v}}' = [1, 0, 0, \operatorname{sign}(A^*(\tilde{\mathbf{v}})), 0, 0]^\top$  where  $\operatorname{sign}(A^*(\tilde{\mathbf{v}}))$  determines the orientation of the triangle. We obtain  $A^*(\tilde{\mathbf{v}}') = \frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{2}$  and  $D_o(\tilde{\mathbf{v}}') = 2$ , thus the coefficients of the depressed quartic equation (E.15) become:

$$p = -\frac{32A_o - 8}{A_o} \quad (\text{E.33})$$

$$q = \frac{64}{A_o} \quad (\text{E.34})$$

$$r = \frac{256A_o + 128}{A_o}. \quad (\text{E.35})$$

Substituting these coefficients in Cardano's formula we obtain:

$$Q_1 = -\left(\frac{32A_o + 4}{3A_o}\right)^2 \quad (\text{E.36})$$

$$Q_2 = \left(\frac{32A_o + 4}{3A_o}\right)^3, \quad (\text{E.37})$$

848 making  $\sqrt{Q_1^3 + Q_2^2} = 0$  and the real root  $\alpha_o$  of Cardano's resolvent cubic to become:

$$\alpha_o = 2\left(\frac{32A_o + 4}{3A_o}\right) + \frac{32A_o - 8}{3A_o} = 32. \quad (\text{E.38})$$



849 We finally use  $\alpha_o$  to extract the roots of the depressed quartic:

$$\lambda = \frac{s_1 \sqrt{32} \lambda_o + s_2 \sqrt{8 \left( \frac{1-s_1}{A_o} \right)}}{\sqrt{2}} \quad (\text{E.39})$$

850 where  $s_1, s_2 \in \{-1, 1\}$ . We thus have the following roots:

$$\lambda \in \left\{ -4, -4, 4 - 2i \sqrt{\frac{2}{A_o}}, 4 + 2i \sqrt{\frac{2}{A_o}} \right\}. \quad (\text{E.40})$$

851 Considering only the real roots we have  $\lambda = -4$ .

852 We now turn to the reverse implication:  $F_2 \Leftarrow A(\tilde{\mathbf{v}}) \neq 0$  and  $\lambda = -4$ . We substitute  
853  $\lambda = -4$  in equation (E.13), giving:

$$-4s \text{sign}(A^*(\tilde{\mathbf{v}}))A(\tilde{\mathbf{v}}) - D_o(\tilde{\mathbf{v}}) = 0. \quad (\text{E.41})$$

854 Since  $A(\tilde{\mathbf{v}}) \neq 0$  and  $D_o(\tilde{\mathbf{v}}) > 0$  then equation (E.41) can only be solved when  
855  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$ . We perform the same similarity transformation used at the begin-  
856 ning of the proof to the unknown input triangle  $\tilde{\mathbf{v}}$  giving  $\tilde{\mathbf{v}}' = [1, 0, x'_b, y'_b, 0, 0]^\top$  where  
857 one of the vertices remains unknown. We then have  $A^*(\tilde{\mathbf{v}}') = \frac{y'_b}{2}$  or  
858  $\text{sign}(A^*(\tilde{\mathbf{v}}'))A(\tilde{\mathbf{v}}') = \frac{y'_b}{2}$  and  $D_o(\tilde{\mathbf{v}}') = x_b'^2 + y_b'^2 + 1$ . which we substitute in equa-  
859 tion (E.41) and obtain:

$$x_b'^2 + y_b'^2 + 2sy'_b + 1 = 0, \quad (\text{E.42})$$

860 which we rewrite as:

$$x_b'^2 + (y'_b + s)^2 = 0. \quad (\text{E.43})$$

861 This is the equation of a single point, making  $\tilde{\mathbf{v}}'$  a right isosceles triangle  $\tilde{\mathbf{v}}' =$   
862  $[0, 1, 0, -s, 0, 0]^\top$  where  $s$  determines the orientation of the triangle. Since  $\tilde{\mathbf{v}}'$  was  
863 a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\tilde{\mathbf{v}}$  is also a right isosceles triangle when  
864  $\text{sign}(A^*(\tilde{\mathbf{v}})) = -s$ , which corresponds to  $F_2$ .  $\square$

*Proof of Lemma 11.* In  $F_3$ ,  $\tilde{\mathbf{v}}$  represents a right isosceles triangle with  $A(\tilde{\mathbf{v}})/4 \geq A_o$   
and no orientation inversion. This implies  $A^*(\tilde{\mathbf{v}}) \neq 0$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ . We  
perform the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma 7, thus the coefficients  
of the depressed quartic equation (30) become:

$$p = -\frac{32A_o + 8}{A_o} \quad (\text{E.44})$$

$$q = \frac{32}{A_o} \quad (\text{E.45})$$

$$r = \frac{256A_o - 128}{A_o}. \quad (\text{E.46})$$

Substituting these coefficients in Cardano's formula we obtain:

$$Q_1 = - \left( \frac{32A_o - 4}{3A_0} \right)^2 \quad (\text{E.47})$$

$$Q_2 = - \left( \frac{32A_o - 4}{3A_0} \right)^3, \quad (\text{E.48})$$

865 making  $\sqrt{Q_1^3 + Q_2^2} = 0$ . After some factoring we obtain the real root  $\alpha_o$  of Cardano's  
866 resolvent cubic as:

$$\alpha_o = -2 \sqrt[3]{ \left( \frac{32A_o - 8sA^*(\tilde{\mathbf{v}}')}{3A_o} \right)^3 + \frac{32A_o + 16sA^*(\tilde{\mathbf{v}}')}{3A_o} }. \quad (\text{E.49})$$

867 Since  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  and  $A^*(\tilde{\mathbf{v}}') = \text{sign}(A^*(\tilde{\mathbf{v}}))A(\tilde{\mathbf{v}}')$ , then  $\alpha_o \in \mathbb{R}$  only when  
868  $A(\tilde{\mathbf{v}}')/4 \geq A_o$ . Under this condition, we obtain  $\alpha_o = 32$ . Substituting in equation  
869 (37) we obtain:

$$\lambda = \frac{s_1 \sqrt{32} \lambda_o + s_2 \sqrt{8 \left( \frac{1-s_1}{A_o} \right)}}{\sqrt{2}} \quad (\text{E.50})$$

870 where  $s_1, s_2 \in \{-1, 1\}$ . We thus have the following roots:

$$\lambda \in \left\{ -4 - 2\sqrt{\frac{2}{A_o}}, -4 + 2\sqrt{\frac{2}{A_o}}, 4, 4 \right\}^\top. \quad (\text{E.51})$$

871 Thus, we have two roots where  $|\lambda| \neq 4$  (which correspond to solutions for Case I)  
872 and there exists at least one solution  $\lambda = 4$  for  $F_3$  (which correspond to the solution  
873 of Case II).  $\square$

874 *Proof of Lemma 12.* We substitute  $\lambda = 4$  in equation (E.13), giving:

$$4s \text{sign}(A^*(\tilde{\mathbf{v}}))A(\tilde{\mathbf{v}}) - \sigma^2(\tilde{\mathbf{v}}) = 0. \quad (\text{E.52})$$

875 Since  $A(\tilde{\mathbf{v}}) \neq 0$  then equation (E.52) can only be solved when  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ .  
876 We perform the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma 8, substitute  
877  $\text{sign}(A^*(\tilde{\mathbf{v}}'))A(\tilde{\mathbf{v}}')$  and  $\sigma^2(\tilde{\mathbf{v}}')$  in equation (E.52) and obtain:

$$x_b'^2 + y_b'^2 - 2sy_b' + 1 = 0, \quad (\text{E.53})$$

878 which we rewrite as:

$$x_b'^2 + (y_b' - s)^2 = 0. \quad (\text{E.54})$$

879 This is the equation of a single point making  $\tilde{\mathbf{v}}'$  a right isosceles triangle  $\tilde{\mathbf{v}}' =$   
880  $[0, 1, 0, s, 0, 0]^\top$  where  $s$  determines the orientation of the triangle. Since  $\tilde{\mathbf{v}}'$  was  
881 a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\tilde{\mathbf{v}}$  is also a right isosceles triangle when  
882  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  for any value  $A(\tilde{\mathbf{v}})$  (including  $A(\tilde{\mathbf{v}})/4 \geq A_o$ ) which corresponds  
883 to  $F_3$ .  $\square$

884 *Proof of Lemma 13.* We perform the same similarity transformation to  $\tilde{\mathbf{v}}$  as in lemma  
885 8 and obtain  $A^*(\tilde{\mathbf{v}}') = \frac{\text{sign}(A^*(\tilde{\mathbf{v}}))}{2}$ . We relax the condition of  $F_3$  where  $\frac{A(\tilde{\mathbf{v}}')}{4} \geq A_o$  by  
886 reformulating  $A_o = \frac{1}{2z}$  where  $z$  is a scaling factor  $z > 0 \in \mathbb{R}$ . We substitute this in  
887 equation (E.51) and extract the roots :

$$\lambda = [-4 - 4\sqrt{z}, -4 + 4\sqrt{z}, 4, 4]^\top. \quad (\text{E.55})$$

888 We start with the solution given by Case I where  $|\lambda| \neq 4$ . We compact both values  
889 as  $\lambda = -4 + s_3 4\sqrt{z}$  where  $s_3 \in \{-1, 1\}$ . We substitute this in equation (E.6) and  
890 obtain:

$$\mathbf{u}' = \left[ \frac{s_3}{\sqrt{z}} \quad 0 \quad 0 \quad \frac{s_3}{\sqrt{z}} \right]^\top. \quad (\text{E.56})$$

891 We calculate the cost of the solution of Case I by substituting  $\mathbf{v}' = [\mathbf{u}', \tilde{x}_c, \tilde{y}_c]$  and  $\tilde{\mathbf{v}}'$   
892 in equation (3) and obtain:

$$\mathcal{C}_1(\mathbf{v}') = \frac{2(\sqrt{z} - s_3)^2}{z} \quad (\text{E.57})$$

893 Now we turn to the solution given by Case II where  $\lambda = 4$ . We calculate the basis  
894 vector  $\mathbf{u}'_c$ , the particular solution  $\mathbf{u}'_p$  and substitute them in equation (E.29) and  
895 obtain:

$$\mathbf{v}' = \left[ \frac{1}{2} \quad \frac{\sqrt{z-4}}{2\sqrt{z}} \quad \frac{\sqrt{z-4}}{2\sqrt{z}} \quad \frac{1}{2} \quad 0 \quad 0 \right]^\top. \quad (\text{E.58})$$

896 We calculate the cost of the solution of Case II by substituting this and  $\tilde{\mathbf{v}}'$  in equation  
897 (3) and obtain:

$$\mathcal{C}_2(\mathbf{v}') = 1 - \frac{2}{z} \quad (\text{E.59})$$

898 We compare the cost of both solutions  $\mathcal{C}_1(\mathbf{v}') \geq \mathcal{C}_2(\mathbf{v}')$  and obtain:

$$\frac{2(\sqrt{z} - s_3)^2}{z} \geq 1 - \frac{2}{z} \quad (\text{E.60})$$

899 After some minor manipulations we obtain:

$$\frac{z - 4s_3\sqrt{z} + 4}{2z} \geq 0 \quad (\text{E.61})$$

900 We substitute  $\sqrt{z} = a$  where  $a > 0 \in \mathbb{R}$  and obtain the quadratic expression:

$$\frac{(a - 4s_3)^2}{2a^2} \geq 0 \quad (\text{E.62})$$

901 which represents an upward opening parabola which is always positive for any value  
 902 of  $a$  and thus  $z$ . This implies that  $\mathcal{C}_1(\mathbf{v}') \geq \mathcal{C}_2(\mathbf{v}')$  for any value  $z$  including  $z > 4$ .  
 903 Since  $\tilde{\mathbf{v}}'$  was a similarity transformation of  $\tilde{\mathbf{v}}$ , then  $\mathcal{C}_1(\mathbf{v}) \geq \mathcal{C}_2(\mathbf{v})$ . This means  
 904 that in  $F_3$  the solution provided by case II has the lowest cost, thus is the optimal  
 905 solution.  $\square$

906 *Proof of Lemma 14.* We start with the forward implication:  $\mathbf{u}_p = 0 \Rightarrow \mathbf{u}'$  is a pair  
 907 of perpendicular vectors of equal length. From equation (E.23), we have that  $\mathbf{u}_p =$   
 908  $0$  when  $\tilde{\mathbf{u}}' = 0$  (trivial solution) or when  $\tilde{\mathbf{u}}' = 0$  is in the kernel of  $X^\dagger$ . From  
 909 the properties of the pseudo-inverse we have that  $\ker(X^\dagger) = \ker(X^\top)$ . Since  $X$   
 910 is symmetric then  $\ker(X^\dagger) = \ker(X) = \mathbf{u}_h$ , as given in equation (E.22). Recall  
 911 from Appendix E.1.3 that the system  $X\mathbf{u}' = \tilde{\mathbf{u}}'$  is solvable if and only if  $\tilde{\mathbf{v}}' =$   
 912  $[\tilde{\mathbf{u}}', 0, 0]$  represents a right isosceles triangle of orientation  $\text{sign}(A^*(\tilde{\mathbf{v}}')) = s \text{sign}(\lambda)$   
 913 or colocated vertices. Still from Appendix E.1.3, we have that  $\mathbf{v}_h = [\mathbf{u}_h, 0, 0]$  is a  
 914 right isosceles triangle of orientation  $\text{sign}(A^*(\mathbf{v}_h)) = -s \text{sign}(\lambda)$ , which contradicts  
 915 the previous statement. This leaves us with  $\mathbf{u}_p = 0$  only when  $\tilde{\mathbf{u}}' = 0$ , which  
 916 represents a set of colocated points centred in the origin. We then calculate  $\mathbf{u}'$  from  
 917 equation (E.25) and obtain:

$$\mathbf{u}' = \begin{bmatrix} -\beta_2 \\ \beta_1 \\ \beta_1 \\ \beta_2 \end{bmatrix}. \quad (\text{E.63})$$

We can then easily show that the vertices distances with the origin are equal, and thus that  $\mathbf{v}'$  is an isosceles triangle:

$$D_{ab} = \sqrt{\beta_1^2 + \beta_2^2}$$

$$D_{ac} = \sqrt{\beta_1^2 + \beta_2^2}.$$

918 Also we can show that  $\mathbf{v}'$  is a right isosceles triangle by calculating the inner product  
 919 of its vertices:

$$I_{ab} = (x'_a \times x'_b) + (y'_a \times y'_b) = 0. \quad (\text{E.64})$$

920 We now turn to the reverse implication:  $\mathbf{u}_p = 0 \neq \mathbf{v}' = [\mathbf{v}', 0, 0]$  is a right isosceles  
 921 triangle, for which we simply provide a counterexample to the positive implication.

922 We use the right isosceles triangle  $\tilde{\mathbf{v}}' = [1, 0, 0, \text{sign}(A^*(\tilde{\mathbf{v}}), 0, 0)]^\top$  where  $A(\tilde{\mathbf{v}}')/4 = A_o$   
 923 and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$ . We calculate its particular solution with equation (E.23) and  
 924 obtain:

$$\mathbf{u}_p = \begin{bmatrix} \frac{1}{2} \\ 0 \\ 0 \\ \frac{1}{2} \end{bmatrix}, \quad (\text{E.65})$$

which implies that  $\mathbf{u}_p \neq 0$ . We substitute  $A(\tilde{\mathbf{v}}')/4 = A_o$  and  $\text{sign}(A^*(\tilde{\mathbf{v}})) = s$  in the  
 constraint equation (E.25) and obtain  $\beta_1^2 = -\beta_2^2$ . This implies that the constraint  
 is fulfilled when  $\beta_1 = \beta_2 = 0$  thus  $\mathbf{u}' = \mathbf{u}_p$ . We easily show that the inter-vertex  
 distances are equal:

$$D_{ab} = \sqrt{(x'_a - x'_b)^2 + (y'_a - y'_b)^2} = \frac{1}{2}$$

$$D_{ac} = \sqrt{(x'_b - x'_c)^2 + (y'_b - y'_c)^2} = \frac{1}{2},$$

925 and also that the vertices are perpendicular by calculating their inner product:

$$I_{ab} = (x'_a \times x'_b) + (y'_a \times y'_b) = 0. \quad (\text{E.66})$$

926 Thus,  $\mathbf{v}$  is a right isosceles triangle despite  $\mathbf{u}_p \neq 0$ . □

927 *Proof of Lemma 15.* Our proof shows that the cost is invariant to the value chosen  
 928 for angle  $\theta$ . We start by translating the coordinate system to bring the fixed vertex  
 929 to the origin as  $\tilde{\mathbf{v}}' = \tilde{\mathbf{v}} - v_c$ , which also translates the unknown vertices to  $\mathbf{v}' = \mathbf{v} - v_c$ .  
 930 Thus, the cost becomes the translated least-squares displacement cost:

$$\mathcal{E}'(\mathbf{v}') = \|\mathbf{v}' - \tilde{\mathbf{v}}'\|^2. \quad (\text{E.67})$$

931 We then continue with the case where  $\tilde{\mathbf{v}}$  is a single point. This implies  $A(\tilde{\mathbf{v}}) = 0$  and  
 932  $k = -1$ . The particular solution  $\mathbf{v}'_p$  can be verified to vanish, from equation (E.23).  
 933 We calculate the basis vector  $\mathbf{v}'_c$  from equation (E.28), leading, from equation (E.29),  
 934 to:

$$\mathbf{u}' = \begin{bmatrix} -\rho \sin(\theta) \\ \rho \cos(\theta) \\ \rho \cos(\theta) \\ \rho \sin(\theta) \end{bmatrix}. \quad (\text{E.68})$$

935 We calculate the cost by substituting  $\tilde{\mathbf{u}}$  and  $\tilde{\mathbf{u}}'$  in equation (E.67):

$$\mathcal{E}'(\mathbf{v}') = 2\rho^2(\sin^2(\theta) + \cos^2(\theta)), \quad (\text{E.69})$$

936 which simplifies to  $\mathcal{C}'(\mathbf{v}') = 2\rho^2$  and is thus independent of  $\theta$ .

937 We continue with the case where  $\tilde{\mathbf{v}}$  is a right isosceles triangle. This implies that  
 938  $A(\tilde{\mathbf{v}}) \neq 0$  and  $k = s \operatorname{sign}(A^*(\tilde{\mathbf{v}}))$ . We perform the same similarity transformation to  
 939  $\tilde{\mathbf{v}}$  as in lemma 10 and obtain  $A^*(\tilde{\mathbf{v}}') = \frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}))}{2}$ . We calculate the basis vector  $\mathbf{u}'_c$   
 940 from equation (E.28), the particular solution  $\mathbf{u}'_p$  from equation (E.23) and substitute  
 941 them in equation (E.29), leading to:

$$\mathbf{u}' = \begin{bmatrix} \frac{1}{2} - \rho \sin(\theta) \\ \rho \cos(\theta) \\ \rho \cos(\theta) \\ \frac{1}{2} + \rho \sin(\theta) \end{bmatrix}. \quad (\text{E.70})$$

942 We calculate the cost by substituting  $\mathbf{v}'$  and  $\tilde{\mathbf{v}}'$  in equation (E.67):

$$\mathcal{C}'(\mathbf{v}') = \frac{1}{2} + 2\rho^2(\sin^2(\theta) + \cos^2(\theta)), \quad (\text{E.71})$$

943 which simplifies to  $\mathcal{C}'(\mathbf{v}') = \frac{1}{2} + 2\rho^2$  and is thus independent of  $\theta$ . □

#### 944 *Appendix E.1.5. Numerical Implementation*

945 We use the theory developed in the previous sections to construct a numerically  
 946 robust procedure, given in Algorithm 8, to solve OTPPAO with one fixed vertex.  
 947 It uses the input vertices  $\tilde{\mathbf{v}}$ , prescribed area  $A_o$  and orientation  $s$  as inputs. It also  
 948 uses an area error tolerance  $E$  to handle round-off in the area constraint (8). Similar  
 949 to Algorithm 1, Algorithm 8 starts by generating the solutions from Case I, then  
 950 Case II, and chooses the optimal one. For Case I, we obtain a list  $\mathbf{v}_1$  of at most 4  
 951 solutions. For Case II, we obtain a single best solution  $\mathbf{v}_2$ , the optimally rotated one,  
 952 the vertices basis and translation to generate all solutions following equation (E.29).  
 953 The overall optimal solution  $\mathbf{v}_o$  is chosen amongst  $\mathbf{v}_1$  and  $\mathbf{v}_2$ . The algorithm returns  
 954 the optimal solution, along with all the solutions from Case I and Case II.

#### 955 *Appendix E.2. Two Fixed Vertices*

956 We assume that  $v_b$  and  $v_c$  are fixed. Thus, we redefine  $\mathbf{v} = [v_a, \tilde{x}_b, \tilde{y}_b, \tilde{x}_c, \tilde{y}_c]^\top$   
 957 where  $v_a$  is the moving vertex. We take  $\frac{\partial \mathcal{L}}{\partial v_a} = 0$  which are essentially the first  
 958 two equalities of equation (12). We rearrange these equations into a set of linear  
 959 equations:

$$\begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} \tilde{x}_a - \frac{s\lambda}{4}(\tilde{y}_c - \tilde{y}_b) \\ \tilde{y}_a - \frac{s\lambda}{4}(\tilde{x}_b - \tilde{x}_c) \end{bmatrix}. \quad (\text{E.72})$$

---

**Algorithm 8** Optimal Triangle Projection with a Prescribed Area and Orientation with a Fixed Vertex

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\mathbf{v}_o$  - optimal triangle,  $\mathbf{v}_1$  - Case I triangle set,  $\mathbf{v}_2$  - Case II optimal triangle,  $\mathbf{u}_c, \mathbf{v}_t$  - Case II basis and offset

- 1: **function** OTTPAO1( $\tilde{\mathbf{v}}, A_o, s, E = 10^{-3}$ )
  - 2:      $\mathbf{v}_1 \leftarrow \text{SOLVECASE1ONEFIXEDVERTEX}(\tilde{\mathbf{v}}, A_o, s, E)$       $\triangleright$  Case I solutions
  - 3:      $(\mathbf{v}_2, \mathbf{v}_c, \mathbf{u}_c, \mathbf{v}_t) \leftarrow \text{SOLVECASE2ONEFIXEDVERTEX}(\tilde{\mathbf{v}}, A_o, s, E)$       $\triangleright$  Case II solutions
  - 4:      $\mathbf{v}_o \leftarrow \text{FINDTRIANGLEOFMINIMALCOST}(\mathbf{v}_1 \cup \{\mathbf{v}_2\})$       $\triangleright$  Optimal solution
  - 5:     **return**  $\mathbf{v}_o, \mathbf{v}_1, \mathbf{v}_2, \mathbf{u}_c, \mathbf{v}_t$
  - 6: **end function**
- 

We thus solve system (E.72) with two cases. In Case I, which is the most general, we have  $v_b \neq v_c$ . In Case II, we have  $v_b = v_c$ . The solution for the latter case is trivial, since no matter what value we give to  $\lambda$ , the non-fixed vertex remains the same ( $x_a = \tilde{x}_a$  and  $y_a = \tilde{y}_a$ ). For Case I, we substitute the result of equation (E.72) in  $\mathbf{v}$  and calculate the signed area constraint (8). The resulting linear equation in  $\lambda$  is  $a_1\lambda + a_0 = 0$  where:

$$\begin{aligned} a_0 &= 4s((\tilde{x}_a - \tilde{x}_c)(\tilde{y}_b - \tilde{y}_a) - (\tilde{x}_a - \tilde{x}_b)(\tilde{y}_c - \tilde{y}_a)) - 8A_o \\ a_1 &= -(\tilde{x}_b^2 + \tilde{x}_c^2 + \tilde{y}_b^2 + \tilde{y}_c^2 - 2\tilde{x}_b\tilde{x}_c - 2\tilde{y}_b\tilde{y}_c). \end{aligned}$$

960 We can rewrite these coefficients more compactly. Concretely,  $a_0$  contain the area  
 961  $A^*(\tilde{\mathbf{v}})$  of the input vertices and  $a_1$  contains the square distance  $P_o$  between the two  
 962 fixed vertices:

$$P_o = (\tilde{x}_b - \tilde{x}_c)^2 + (\tilde{y}_b - \tilde{y}_c)^2. \quad (\text{E.73})$$

963 We solve for  $\lambda$  and obtain:

$$\lambda = 8 \frac{sA^*(\tilde{\mathbf{v}}) - A_o}{P_o}. \quad (\text{E.74})$$

964 We substitute  $\lambda$  from equation (E.74) in equation (E.72) and obtain:

$$\begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} \tilde{x}_a \\ \tilde{y}_a \end{bmatrix} - 4 \frac{sA_o + A^*(\tilde{\mathbf{v}})}{P_o} \begin{bmatrix} (\tilde{y}_c - \tilde{y}_b) \\ (\tilde{x}_b - \tilde{x}_c) \end{bmatrix}. \quad (\text{E.75})$$

965 The numerically robust procedure of this solution is given in Algorithm 11.

---

**Algorithm 9** Closed-form Analytic Solution to Case I of OTPPAO with One Fixed Vertex

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\nu_1$  - solution list

```

1: function SOLVECASE1ONEFIXEDVERTEX( $\tilde{\mathbf{v}}, A_o, s, E$ )
2:    $D_o(\tilde{\mathbf{v}}) \leftarrow (\tilde{x}_a - \tilde{x}_c)^2 + (\tilde{y}_a - \tilde{y}_c)^2 + (\tilde{x}_b - \tilde{x}_c)^2 + (\tilde{y}_b - \tilde{y}_c)^2$ 
3:    $p \leftarrow -\frac{16(2A_o + sA^*(\tilde{\mathbf{v}}))}{A_o}$        $\triangleright$  Compute the coefficients of the depressed quartic
4:    $q \leftarrow \frac{32D_o(\tilde{\mathbf{v}})}{A_o}$ 
5:    $r \leftarrow \frac{256(A_o - sA^*(\tilde{\mathbf{v}}))}{A_o}$ 
6:    $\boldsymbol{\lambda} \leftarrow \text{FERRARISOLUTION}(p, q, r)$        $\triangleright$  Solve for the four possible Lagrange
      multipliers
7:    $\nu_1 \leftarrow \emptyset$        $\triangleright$  Create an empty set of solutions
8:   for  $t \leftarrow 1, \dots, 4$  do       $\triangleright$  Generate and select the triangles
9:      $\lambda \leftarrow \text{Re}(\boldsymbol{\lambda}(t))$        $\triangleright$  Keep the real part
10:     $h \leftarrow (\lambda^2 - 16)^\dagger$        $\triangleright$  Compute the inverse denominator
11:     $\mathbf{u} \leftarrow h \begin{bmatrix} \tilde{x}_c \lambda^2 + 4s(\tilde{y}_b - \tilde{y}_c)\lambda - 16\tilde{x}_a \\ \tilde{y}_c \lambda^2 + 4s(\tilde{x}_c - \tilde{x}_b)\lambda - 16\tilde{y}_a \\ \tilde{x}_c \lambda^2 + 4s(\tilde{y}_c - \tilde{y}_a)\lambda - 16\tilde{x}_b \\ \tilde{y}_c \lambda^2 + 4s(\tilde{x}_a - \tilde{x}_c)\lambda - 16\tilde{y}_b \end{bmatrix}$        $\triangleright$  Compute the vertices
12:     $\mathbf{v} = [\mathbf{u}, \tilde{x}_c, \tilde{y}_c]$ 
13:    if  $|sA^*(\mathbf{v}) - A_o| \leq E$  then       $\triangleright$  Check the area constraint
14:       $\nu_1 \leftarrow \nu_1 \cup \{\mathbf{v}\}$        $\triangleright$  Add the vertices to the solution set
15:    end if
16:  end for
17:  return  $\nu_1$ 
18: end function

```

---



---

**Algorithm 10** Closed-form Analytic Solution to Case II of OTPPAO with One Fixed Vertex

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - area error tolerance

**Output:**  $\mathbf{v}_2$  - optimal triangle,  $\mathbf{u}_c, \mathbf{v}_t$  vertices basis and translation.

```

1: function SOLVECASE2ONEFIXVERT( $\tilde{\mathbf{v}}, A_o, s, E$ )
2:   if  $A(\mathbf{v}) \leq E$  then                                     ▷ Check the input's area
3:      $k \leftarrow s \operatorname{sign}(A^*(\mathbf{v}))$                        ▷ Compute  $k$  for a right isosceles triangle
4:   else
5:      $k \leftarrow -1$                                            ▷ Compute  $k$  for a single point
6:   end if
7:    $\tilde{\mathbf{v}}' \leftarrow \tilde{\mathbf{v}} - v_c$                                    ▷ Translate the input vertices
8:    $\rho \leftarrow \sqrt{\frac{\operatorname{sign}(A^*(\tilde{\mathbf{v}}'))A^*(\tilde{\mathbf{v}}')-4kA_o}{2}}$    ▷ Computes the area constraint parameter
9:    $\mathbf{u}_c \leftarrow \operatorname{Re}(\rho) [0 \ ks \ 1 \ 0]^\top$                  ▷ Compute the solution basis
10:   $\mathbf{u}_p \leftarrow \frac{1}{4} \begin{bmatrix} \tilde{x}'_a + \operatorname{sign}(A^*(\tilde{\mathbf{v}}))\tilde{y}'_b \\ \tilde{y}'_a - \operatorname{sign}(A^*(\tilde{\mathbf{v}}))\tilde{x}'_b \\ \tilde{x}'_b - \operatorname{sign}(A^*(\tilde{\mathbf{v}}))\tilde{y}'_a \\ \tilde{y}'_b + \operatorname{sign}(A^*(\tilde{\mathbf{v}}))\tilde{x}'_a \end{bmatrix}$    ▷ Compute the particular solution
11:   $\tilde{U}_2 \leftarrow$  rearrange  $\tilde{\mathbf{u}}'$  into a  $2 \times 2$  matrix
12:   $U_c \leftarrow$  rearrange  $\mathbf{u}_c$  into a  $2 \times 2$  matrix
13:   $(U_1, \Sigma, U_2) \leftarrow \operatorname{SVD}(\tilde{U}_2 U_c^\top)$                  ▷ Compute the optimal rotation
14:   $D \leftarrow \operatorname{diag}(1, \det(U_2 U_1^\top))$ 
15:   $R \leftarrow U_2 D U_1^\top$ 
16:   $\mathbf{v}_t \leftarrow [\mathbf{u}_p \ 0 \ 0]^\top + [v_c \ v_c \ v_c]^\top$          ▷ Compute the translation vector
17:   $\mathbf{v}_2 = [\mathcal{R}(\theta)\mathbf{u}_c \ 0 \ 0]^\top + \mathbf{v}_t$                    ▷ Compute the optimal solution
18:  return  $\mathbf{v}_2, \mathbf{u}_c, \mathbf{v}_t$ 
19: end function

```

---

---

**Algorithm 11** Optimal Triangle Projection with a Prescribed Area and Orientation with Two Fixed Vertices

---

**Input:**  $\tilde{\mathbf{v}}$  - input vertices,  $A_o$  - prescribed area,  $s$  - prescribed orientation,  $E$  - distance error tolerance

**Output:**  $\mathbf{v}_o$  - optimal triangle

```
1: function OTTPAO2( $\tilde{\mathbf{v}}, A_o, s, E = 10^{-3}$ )
2:    $P_o = (\tilde{x}_b - \tilde{x}_c)^2 + (\tilde{y}_b - \tilde{y}_c)^2$   $\triangleright$  Compute square distance between fixed vertices
3:   if  $P_o > E$  then
4:      $\begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} \tilde{x}_a \\ \tilde{y}_a \end{bmatrix} - 4P_o^\dagger(sA_o + A^*(\tilde{\mathbf{v}})) \begin{bmatrix} \tilde{y}_c - \tilde{y}_b \\ \tilde{x}_b - \tilde{x}_c \end{bmatrix}$ 
5:   else
6:      $\begin{bmatrix} x_a \\ y_a \end{bmatrix} = \begin{bmatrix} \tilde{x}_a \\ \tilde{y}_a \end{bmatrix}$ 
7:   end if
8:    $\mathbf{v}_o \leftarrow [x_a \ y_a \ \tilde{x}_b \ \tilde{y}_b \ \tilde{x}_c \ \tilde{y}_c]^\top$   $\triangleright$  Compute the optimal solution
9:   return  $\mathbf{v}_o$ 
10: end function
```

---

## 966 References

- 967 [1] S. Sun, M. Zhang, G. Zhihong, Smoothing Algorithm for Planar and Surface  
968 Mesh Based on Element Geometric Deformation, *Mathematical Problems in*  
969 *Engineering* 2015 (2015) 1–9. doi:10.1155/2015/435648.
- 970 [2] D. Vartziotis, T. Athanasiadis, I. Goudas, J. Wipper, Mesh smoothing  
971 using the Geometric Element Transformation Method, *Computer Meth-*  
972 *ods in Applied Mechanics and Engineering* 197 (45) (2008) 3760–3767.  
973 doi:10.1016/j.cma.2008.02.028.
- 974 [3] G. Irving, C. Schroeder, R. Fedkiw, Volume conserving finite element simu-  
975 lations of deformable models, *ACM Transactions on Graphics (TOG)* 26 (3)  
976 (2007) 13–es. doi:10.1145/1276377.1276394.  
977 URL <https://doi.org/10.1145/1276377.1276394>
- 978 [4] N. Abu Rumman, M. Fratarcangeli, Position-Based Skinning for Soft Ar-  
979 ticulated Characters, *Computer Graphics Forum* 34 (6) (2015) 240–250.  
980 doi:10.1111/cgf.12533.  
981 URL <https://doi.org/10.1111/cgf.12533>

- 982 [5] T.-C. Tsai, Position Based Dynamics, in: N. Lee (Ed.), Encyclopedia of Com-  
983 puter Graphics and Games, Springer International Publishing, Cham, 2017, pp.  
984 1–5. doi:10.1007/978-3-319-08234-9-92-1.  
985 URL <https://doi.org/10.1007/978-3-319-08234-9-92-1>
- 986 [6] M. Müller, B. Heidelberger, M. Hennix, J. Ratcliff, Position Based  
987 Dynamics, *J. Vis. Comun. Image Represent.* 18 (2) (2007) 109–118.  
988 doi:10.1016/j.jvcir.2007.01.005.  
989 URL <http://dx.doi.org/10.1016/j.jvcir.2007.01.005>
- 990 [7] J. Bender, M. Müller, M. A. Otaduy, M. Teschner, M. Macklin, A Survey on  
991 Position-Based Simulation Methods in Computer Graphics, *Computer Graphics*  
992 *Forum* 33 (6) (2014) 228–251. doi:10.1111/cgf.12346.  
993 URL <https://doi.org/10.1111/cgf.12346>
- 994 [8] M. Kelager, S. Niebe, K. Erleben, A Triangle Bending Constraint  
995 Model for Position-Based Dynamics, The Eurographics Association, 2010.  
996 doi:<http://dx.doi.org/10.2312/PE/vriphys/vriphys10/031-037>.
- 997 [9] Y. Wang, X. Wu, G. Wang, An Angle Bending Constraint Model for Position-  
998 Based Dynamics, in: 2014 International Conference on Virtual Reality and Vi-  
999 sualization, 2014, pp. 430–434, iSSN: null. doi:10.1109/ICVRV.2014.58.
- 1000 [10] M. Tournier, M. Nesme, B. Gilles, F. Faure, Stable constrained dynam-  
1001 ics, *ACM Transactions on Graphics (TOG)* 34 (4) (2015) 132:1–132:10.  
1002 doi:10.1145/2766969.  
1003 URL <https://doi.org/10.1145/2766969>
- 1004 [11] Y. Gu, K. Yang, C. Lu, Constraint Solving Order in Position Based Dynamics,  
1005 in: Proceedings of the 2017 2nd International Conference on Electrical, Au-  
1006 tomation and Mechanical Engineering (EAME 2017), 2017. doi:10.2991/eame-  
1007 17.2017.46.
- 1008 [12] S. Bouaziz, S. Martin, T. Liu, L. Kavan, M. Pauly, Projective dy-  
1009 namics: Fusing constraint projections for fast simulation Article No.  
1010 154doi:10.1145/2601097.2601116.  
1011 URL <https://repository.upenn.edu/hms/146>
- 1012 [13] J.-P. Tignol, Galois' Theory Of Algebraic Equations, Wspc, Singapore ; River  
1013 Edge, NJ, 2001.

- 1014 [14] G. Cardano, T. R. Witmer, O. Ore, *The Rules of Algebra: Ars Magna*, Courier  
1015 Corporation, 1545, google-Books-ID: ZMZY79bvOPgC.
- 1016 [15] D. Herbison-Evans, *Solving Quartics and Cubics for Graphics*, in: A. W.  
1017 Paeth (Ed.), *Graphics Gems V*, Academic Press, Boston, 1995, pp. 3–15.  
1018 doi:10.1016/B978-0-12-543457-7.50009-7.
- 1019 [16] H. Helfgott, M. Helfgott, *A Modern Vision of the Work of Cardano and Ferrari*  
1020 *on Quartics*, *Convergence* (Feb. 2010).
- 1021 [17] S. L. Shmakov, *A Universal Method of Solving Quartic Equations*, *International*  
1022 *Journal of Pure and Applied Mathematics* 71 (2) (2011) 251–259, publisher:  
1023 Academic Publications, Ltd.  
1024 URL <https://ijpam.eu/contents/2011-71-2/7/index.html>
- 1025 [18] K. Arun, T. Huang, S. Blostein, *Least-squares fitting of two 3-D point sets*, *Pat-*  
1026 *tern Analysis and Machine Intelligence*, *IEEE Transactions on PAMI-9* (1987)  
1027 698 – 700. doi:10.1109/TPAMI.1987.4767965.
- 1028 [19] P.-O. Persson, G. Strang, *A Simple Mesh Generator in MATLAB*, *SIAM Review*  
1029 46 (2) (2004) 329–345. doi:10.1137/S0036144503429121.  
1030 URL <https://epubs.siam.org/doi/abs/10.1137/S0036144503429121>
- 1031 [20] M. Friendly, G. Monette, J. Fox, *Elliptical Insights: Understanding Statisti-*  
1032 *cal Methods through Elliptical Geometry*, *Statistical Science* 28 (Feb. 2013).  
1033 doi:10.1214/12-STS402.