Software Tool for Cranial Orthosis Design

Milan Jaros, Tomas Karasek, Petr Strakos, and Alena Vasatova(✉)

IT4Innovations, VSB - Technical University of Ostrava,
17. listopadu 15/2172, 708 33 Ostrava-Poruba, Czech Republic
{milan.jaros,tomas.karasek,petr.strakos,alena.vasatova}@vsb.cz

Abstract. Cranial orthoses are used to correct an abnormal children head shape, and such they have to be designed individually. Customization of those orthoses is currently fully manual task. A software tool should make this process semi-automatic with only small intervention from the user and speed up the whole process. In the future, this tool will be part of the process chain from 3D scanning of the patient head to the 3D printing of the final product. This will allow to produce the orthosis anywhere, without necessity to have expensive devices on one place. For high quality of 3D printing, 3D computer models with high-resolution meshes must be used. We have started development of our tool by rapid testing of methodology. For this purpose we used open source software Blender. Although Blender's functions we used are more robust, they are also unnecessary computationally more expensive. For this reason we have implemented the necessary transformation functions using radial basis functions (RBF) which can be easily modified to include rigid body movements.

Keywords: Cranial orthosis · Software tool · Blender
Radial basis function · Rigid parts

1 Introduction

Treatment of skull deformities of children by cranial orthosis, see Fig. 1, has been adopted by pediatricians and has been increasingly used since it was first documented in 1979, by [3]. Orthosis could be used to fix positional skull deformities such as plagiocephaly, brachycephaly and scaphocephaly, see Fig. 2. These asymmetric head shapes are caused by external forces applied to an infant's malleable skull during prenatal/postnatal development or at birth (e.g. multiple-birth infants, sleeping position or premature birth). Majority of skull deformities present at birth improve spontaneously approximately six weeks after the delivery. Therefore, the most important deformities are those that develop throughout the first months of child's life.

The number of cranial deformities has increased considerably since international efforts of pediatricians recommended the sleeping supine position as a strategy to reduce sudden death syndrome of the newborns. Keeping infant too long in one position (in car seats, baby carriers and other accessories) has also

T. Kozubek et al. (Eds.): HPCSE 2017, LNCS 11087, pp. 88–100, 2018.
https://doi.org/10.1007/978-3-319-97136-0_7

helped to decrease the time children remain in the prone positions, but on the other hand increases the number of children who can develop cranial deformities. More information can be found in [9,10].

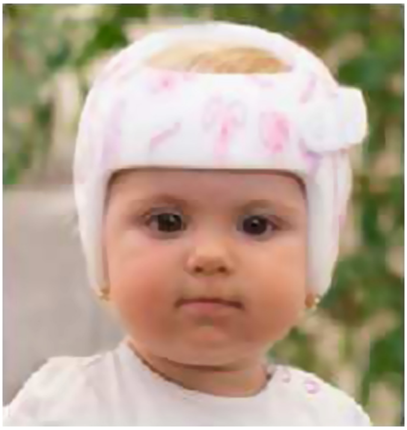


Fig. 1. Example of the cranial orthosis.

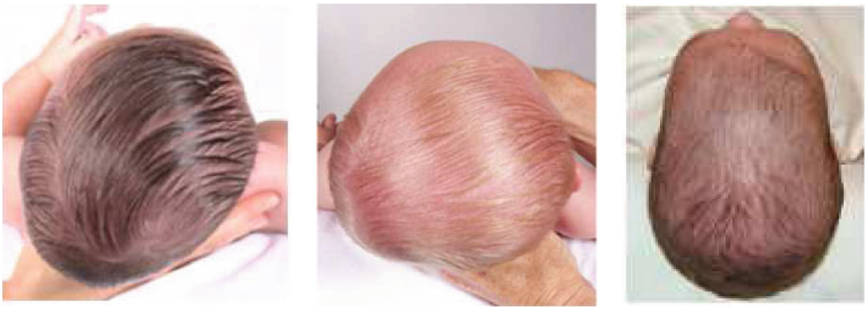


Fig. 2. Examples of plagiocephaly, brachycephaly and scaphocephaly.

When manufactured, every orthosis has to be designed and customized individually, which is currently a fully manual task. Our software tool, which we have developed, requires only small intervention from a medical technician. This speeds up the orthosis production and enables individual parts of the process from 3D scanning to 3D printing to be held anywhere in the world without expensive devices on one place.

To manufacture cranial orthosis by the method of 3D printing, 3D computer models with high-resolution meshes must be used to achieve a product of high quality. To be able to modify the 3D models, large system of linear equations must be solved. This system grows enormously with the size of the used 3D models which requires the use of efficient and fast solvers.

For purpose of rapid prototyping and to test proposed methodology for modification of 3D models of cranial orthosis based on mesh morphing we have used open source 3D graphics and animation software Blender. Our first implementation of the tool was based on modification and enhancement of Blender's two specific modifiers, MeshDeform and ShrinkWrap. Existing methods and functions implemented in MeshDeform were parallelized to improve their performance. However, even parallel version of the MeshDeform did not provide satisfactory performance, and we decided to abandon the functionality of MeshDeform and decided to implement our own transformation procedure using Radial Basis Function (RBF).

RBFs have become a well-established tool to interpolate scattered data. They are used as a transformation functions to interpolate the displacements of the boundary nodes of the mesh to the inner domain. The method requires solving a relatively small system of equations. But the relativity is caused by comparing the number of boundary nodes to the whole mesh. With huge meshes even the boundary becomes large. The implementation of the method is quite simple, even for the 3D applications, because no grid-connectivity information is needed. Parallelization of the concept is also straightforward.

2 Cranial Orthosis Model

During the customization of the generic cranial orthosis, the model is modified to individual patient based on idealized 3D scan of the head, see Fig. 3. This idealized scan is based on the original deformed scan, which is transformed to be as close to ideal head shape as possible. The transformation has to meet certain rules. For example, the circumference of the head can change by maximally 2–3 cm (according the age), the width of the head can not change, the head has to be symmetric etc. This transformation is also done fully manually by technician in CAD like software and it is planned to be automatized. The scan of head is cropped afterwards by outlines specified by the medical technician.

The model itself is composed of two parts: a helmet and a locking mechanism, see Fig. 4. Each part is represented by high-resolution mesh.

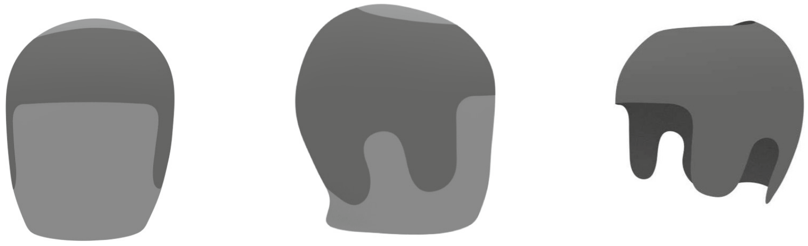
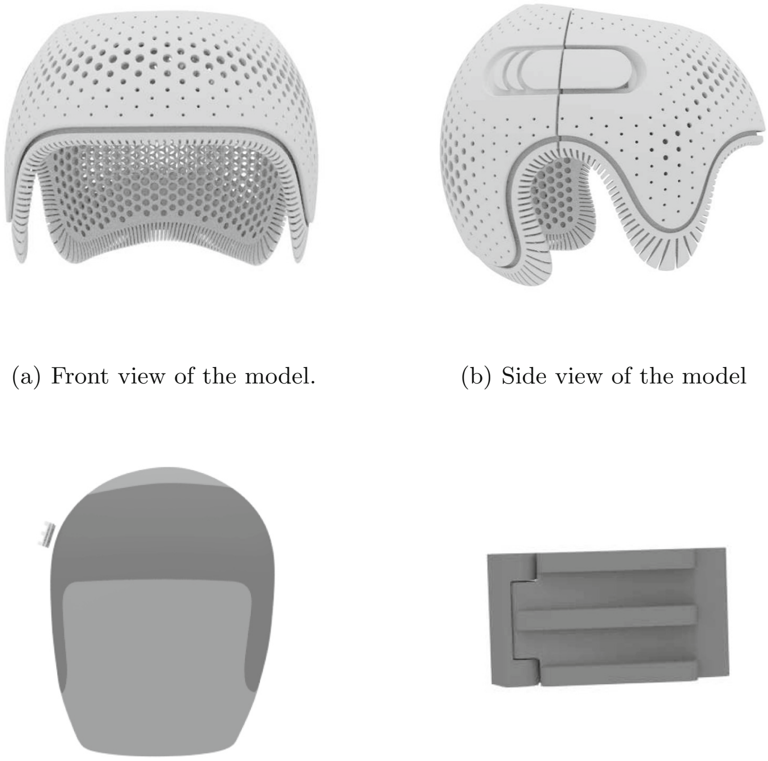


Fig. 3. Input data: original and cropped 3D scan of patient head.



(a) Front view of the model.

(b) Side view of the model

(c) Rigid part (the locking mechanism) from the front.

(d) Rigid part (the locking mechanism) from the side.

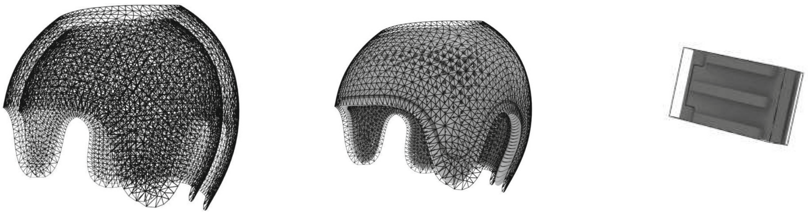
Fig. 4. Input data: cranial orthosis model.

Beside that, we also need auxiliary meshes, so-called cages, see Fig. 5 which purpose is to serve as control points for computing RBF (main cage) and to compute rigid body movement of the locking mechanism (the secondary cage).

The goal of the transformation is a non-rigid deformation of the orthosis body to fit the cropped scan and a rigid transformation of the locking mechanism to its specified position. The locking mechanism can only translate and rotate to preserve its functionality.

3 Morphing Algorithm

At first, we used modification and enhancement of Blender for rapid testing of the proposed methodology. We bound the model mesh with the control cage using MeshDeform modifier, then we shrank the control cage to the scans by Shrinkwrap modifier (the function is based on finding closest point on surface



(a) The main cage.

(b) The main cage with the model.

(c) The secondary cage with the locking mechanism.

Fig. 5. Input data: cages.

with given offset). MeshDeform modifier automatically deforms the model mesh according to the cage deformation, based on [5]. Since this step is computationally extensive MeshDeform modifier was parallelized using MPI technology to improve its speed and to allow handling of large data sets.

Although, parallel version of the MeshDeform modifier gives good results, it is still computationally extensive and we have replaced it completely by transformation using radial basis function (RBF), which is computationally less expensive. Transformation by RBF also allows easy incorporation of the rigid parts, which MeshDeform modifier cannot implicitly do.

The final version of the algorithm can be described as follows:

1. Read input Data.
2. Shrink main cage on cropped scan of the head.
3. Project rigid cage to main cage, find principal axes.
4. Identify local rigid body movement from shrunk and original principal axes.
5. Compute main elastic transformation based on RBF with rigid body movement.
6. Transform given parts.
7. Save results.

Graphical representation of the algorithm in form of simplified example is depicted in Fig. 6.

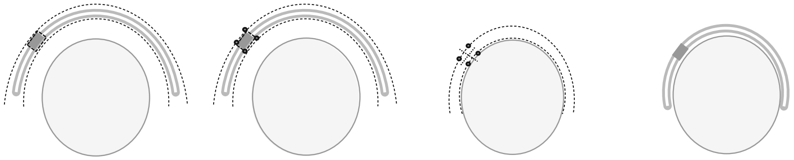


Fig. 6. Simplified example; from left to right - input data, projected rigid cage on the main cage before shrink, Shrunk cages on the scan of the head with principal axes of the locking mechanism, final result

4 Radial Basis Function

Basic principle of RBF could be described in such a way that we are looking for a function $\varphi : \mathbb{R}^3 \to \mathbb{R}^3$ that exactly interpolates the displacement $u_j \in \mathbb{R}^3$ of given control points $x_j \in \mathbb{R}^3$, $j = 1, \ldots, m$ and smoothly interpolates this displacement into the mesh. RBF are well known to be suitable for solving this type of problem [2]. The displacement function is then represented as a linear combination of polynomial corrections p_i and a linear combination of radially symmetric kernels $\rho_{x_j}(x) = \rho(\|x - x_j\|_{\mathbb{R}^3})$ located at centers x_j, independent for each dimension l = 1, 2, 3:

$$\varphi_l(x) = \sum_{i=1}^{n_p} \beta_{l,i} p_i(x) + \sum_{j=1}^{m} \theta_{l,j} \rho_{x_j}(x),$$

where n_p is the dimension of used polynomials. We choose linear, thus $n_p = 4$.

The coefficients $\beta_l \in \mathbb{R}^{n_p}$ and $\theta_l \in \mathbb{R}^m$ are defined by

$$\begin{pmatrix} K & B^\top \\ B & 0 \end{pmatrix} \begin{pmatrix} \theta_l \\ \beta_l \end{pmatrix} = \begin{pmatrix} u_l \\ 0 \end{pmatrix}, \tag{1}$$

where

$$K := (\rho(\|x_j - x_k\|_{\mathbb{R}^3}))_{j,k=1,\ldots,m} \in \mathbb{R}^{m \times m},$$
$$B := (p_i(x_k))_{\substack{i=1,\ldots,n_p \\ k=1,\ldots,m}} \in \mathbb{R}^{n_p \times m}$$

and

$$u_l := (u_{1,l}, \ldots, u_{j,l}) \in \mathbb{R}^m.$$

The choice of the kernel function $\rho(r)$ has a significant influence on the result of the deformation. We use triharmonic [12]

$$\rho(r) = r^3$$

or thin plate spline (TPS) [7]

$$\rho(r) := \frac{\Gamma(3/2 - q)}{2^{2q} \pi^{3/2} (q-1)!} r^{2q-3}.$$

To be able to handle rigid transformation, we incorporated techniques from [6], where certain parts of image are handled as rigid.

Lets have n rigid objects with predefined linear transformations denoted by matrices $L_q \in \mathbb{R}^{3 \times n_p}$, $q = 1, \ldots, n$. Instead of discrete distance transformations used in image processing we compute distance between given point and closest point of the mesh using octree-based spatial algorithm, which can search the mesh to quickly locate the point on the mesh face [11]. $\mathcal{D}_0(x)$ represents the distance from a point x to the closest object and $\mathcal{D}_q(x)$, $q = 1, \ldots, n$ to q-th object. These functions ensure that the non-linear part of the transformation and the linear transformations L_q, $q \neq r$ tend to zero as we move towards the r-th rigid object.

The coefficients β_l for polynomial corrections are replaced by weighted sum of the individual object linear transformations

$$\mathcal{L}(x) = \sum_{q=1}^{n} w_q(x) L_q,$$

where

$$w_q(x) = \frac{v_q(x)}{\sum_{r=1}^{n} v_r(x)},\ v_q(x) = \frac{1}{\mathcal{D}_q(x)^{\mu}}$$

The kernels are also weighted

$$\tilde{\rho}_{x_j}(x) = |\mathcal{D}_0(x)|\,|\mathcal{D}_0(x_j)|\,\rho_{x_j}(x).$$

Thus we get transformation function

$$\varphi_l(x) = \sum_{i=1}^{n_p} \mathcal{L}(x) p_i(x) + \sum_{j=1}^{m} \theta_{l,j} \tilde{\rho}_{x_j}(x),$$

and Eq. (1) can be rewritten as

$$K\theta_l + T = u_l,\ l = 1, 2, 3,$$

where

$$T = \begin{pmatrix} p(x_1)^T \mathcal{L}(x_1)^T \\ p(x_2)^T \mathcal{L}(x_2)^T \\ \vdots \\ p(x_m)^T \mathcal{L}(x_m)^T \end{pmatrix}.$$

Linear transformation matrix L_q consists of 12 coefficients, and we need 4 points in space to determine them. The easy way is to take these points from

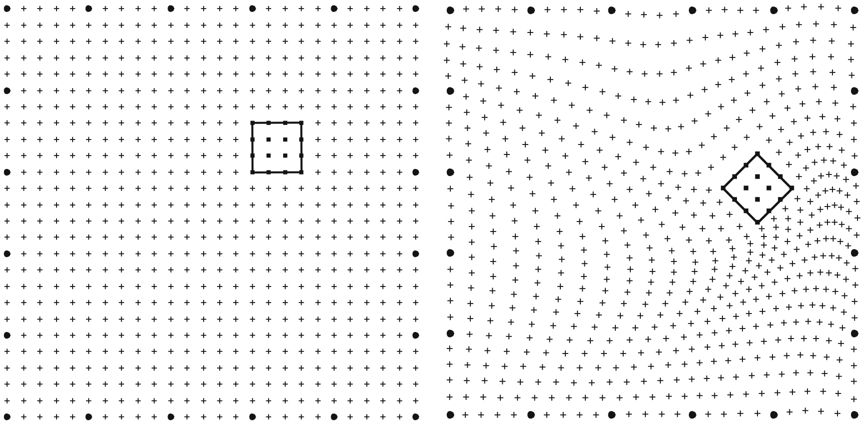


Fig. 7. Simplified 2D example.

principal axes of the rigid cage, Principal axes are obtained through the principal component analysis (PCA). PCA is a statistical method used to estimate the necessary information from the measured data [4,8]. We are able to determine the axes through the use of eigenvalues and eigenvectors of the covariance matrix consisting of a small group of neighboring points.

In Fig. 7 we show the simplified 2D example of rigid object transformation as a part of a more complex mesh. The cage points are marked by circles and the rigid object is marked by square.

5 Implementation

For our implementation we use the VTK library to work with 3D geometry and mesh models. To solve large systems of linear equations we use the Intel MKL library. This library offers direct solvers and it is optimized for high performance.

Beside the time necessary to obtain the solution of the unknowns from the linear system of equations in (1), another time demanding operation in the designing process is the transformation of the orthosis model to the final shape. The model consists of millions of polygons, therefore to spatially transform each of them we have parallelized the operation by utilizing OpenMP technology.

In the following we provide the pseudo code of the whole designing process. It goes as such:

```
1. Read mesh files (Wavefront .obj) - see Figs. 4 and 5.
   (a) the outer part of the main cage for helmet.
   (b) the inner part of the main cage for helmet.
   (c) the secondary cage for the rigid part.
   (d) the model of the helmet.
   (e) the model of the rigid part.
   (f) the model of the patient's head.
2. Shrink cages.
   (a) shrink the outer part of the main cage (1a) to the head (1f) and offset it.
   (b) shrink the inner part of the main cage (1b) to the head (1f).
   (c) shrink the cage of the rigid part (1c) to the inner part of the original main cage (1b).
   (d) shrink the cage of the rigid part (1c) to the shrunk inner part of the main cage (2b).
3 Merge cages.
   (a) merge inner (1b) and outer (1a) part of the original main cage.
   (b) merge inner (2b) and outer (2a) part of the shrunk main cage.
4. Estimate the linear transformation of the rigid part between (2c) and (2d).
   (a) compute the coordination system of the (2c) using PCA.
   (b) compute the coordination system of the (2d) using PCA.
   (c) estimate the linear transformation from the solution of the problem $Ax = b$ which is based on the results of step (4a) and (4b).
5. Compute main elastic transformation based on RBF with rigid body movement.
```

   (a) utilize the above results to assemble another $Ax = b$ problem, detailed description is given in Sect. 4.
   (b) obtain solution by applying direct solver for dense matrices (Intel MKL).
6. Transform model of the orthosis (1d) and (1e).
   (a) utilize solution x from (5b), detailed description is given in Sect. 4.
7. Save results to the OBJ files.

6 Results

To establish performance of our implementation we have performed measurements focusing on algorithm speed and its possible speed-up by utilizing OpenMP framework on multiple cores. We have also measured computational demands of the algorithm based on the model size.

For all the tests, configuration of the RBF and the solver was as follows:

- Thin plate spline (TPS) has been used as a kernel function $\rho(r)$, see Sect. 4.
- Bunch-Kaufman factorization of a symmetric matrix using packed storage has been used as a solver.

6.1 Algorithm Speed and Possible Speed-Up by OpenMP on Multiple CPU Cores

In Table 1 we provide time measurements of different parts of the implemented algorithm. As can be seen, the most time consuming part is the transformation of the model to the desired shape and size. Fortunately, this step can be easily parallelized and more computational resources can be used. Effect of such parallelization is documented in Fig. 8. The problem description that specifies this particular measurement is following:

- Number of vertices from step (3a) of pseudocode: 5452
- Number of vertices from step (3b) of pseudocode: 5452
- Number of vertices from step (1d) of pseudocode: 612546
- Number of vertices from step (1e) of pesudocode: 4818
- System size (size of A from step (5)): 5452 × 5452

Table 1. Computation times within different phases of the algorithm while utilizing OpenMP on 1 to 24 CPU cores

CPU cores [-]	1	2	4	8	16	24
Shrink [s] (2)	0.80	0.75	0.73	0.72	0.73	0.72
Prepare [s] (3)–(4)	85.36	46.44	23.32	12.30	7.96	5.63
Solve [s] (5)	3.94	2.65	2.18	1.96	1.88	1.88
Transform [s] (6)	248.07	124.19	62.29	32.56	15.75	10.87
Total [s] (2)–(6)	338.16	174.03	88.51	47.54	26.33	19.09

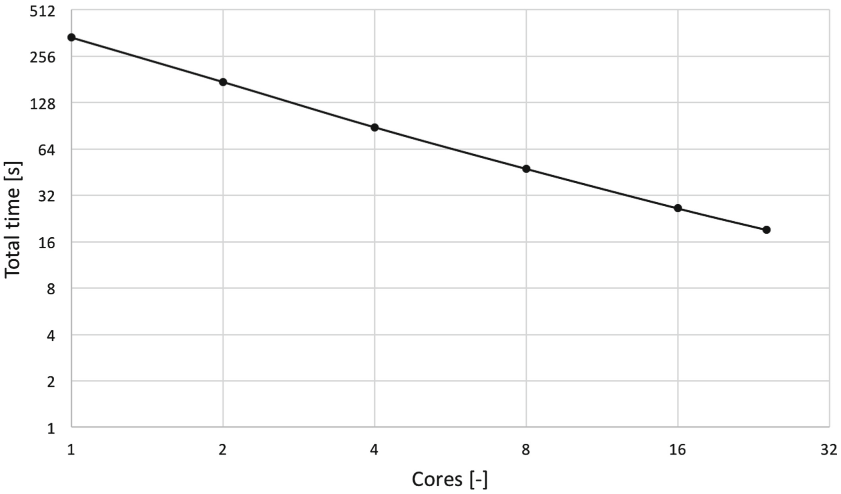


Fig. 8. Total time of the algorithm while utilizing increasing number of CPU cores

Table 2. Comparison of computation times for different sizes of the model

Model size [vertices] (1d)	243897	494826	1033902
Shrink [s] (2)	0.24	0.72	2.61
Prepare [s] (3)–(4)	1.65	4.83	18.54
Solve [s] (5)	0.44	1.89	10.19
Transform [s] (6)	2.76	8.79	38.91
Total [s] (2)–(6)	5.09	16.23	70.25

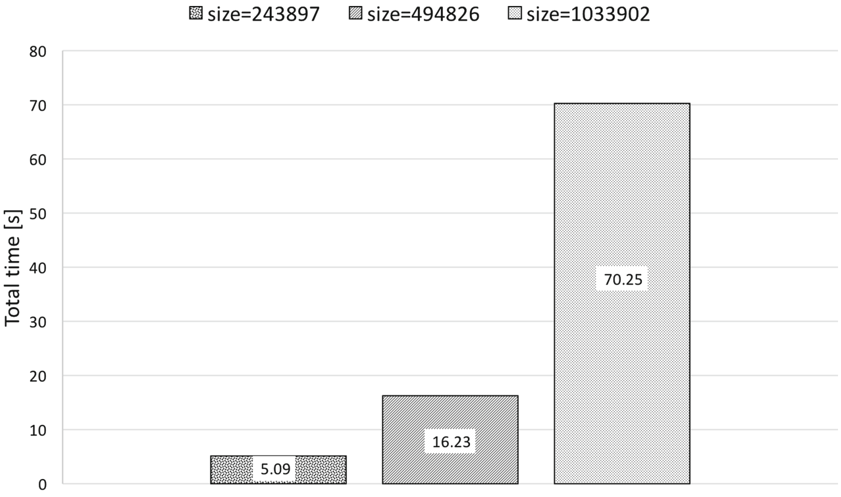


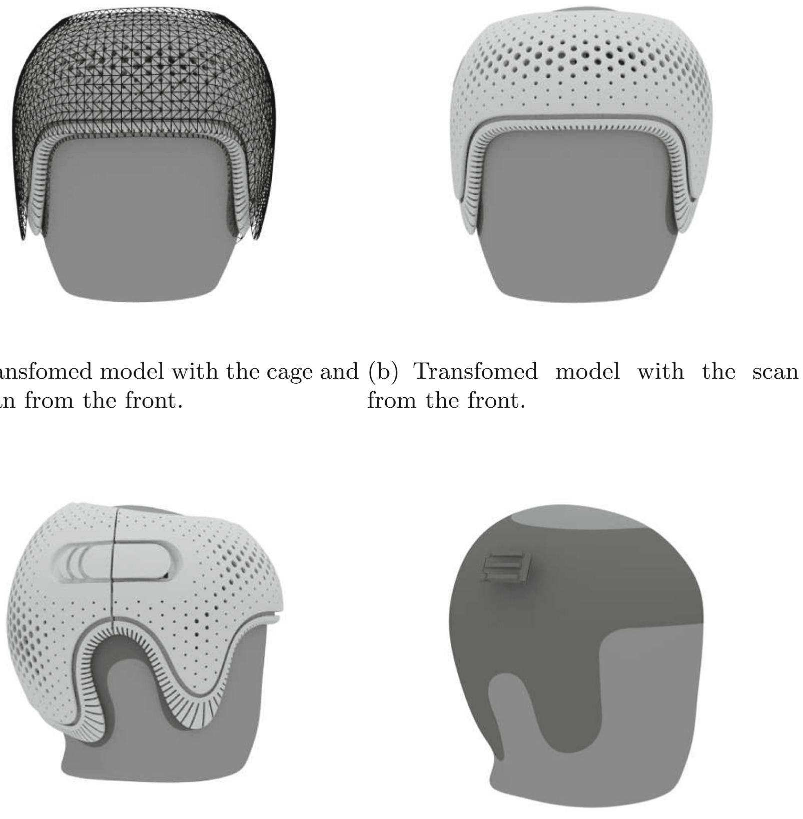
Fig. 9. Total computation time of the algorithm for different sizes of the model

6.2 Computation Times for Different Sizes of the Model

To evaluate algorithm behavior on different sizes of the problem we have tested it on three different models, see Table 2. Within the test, all available resources at computing nodes have been used.

For tests we have always used one node of Salomon supercomputer [1] where each node is equipped with powerful x86-64 computer consisting of 24 cores (2 x Intel Xeon E5-2680v3, 2.5 GHz, 12 cores) and 128 GB RAM.

Total computation times of the algorithm obtained on the three different sizes of the model are well depicted in Fig. 9. The largest model fitting to the desired size and shape of the head is shown in Fig. 10. This particular model also fulfills the required quality demands for the 3D print, which is about 1 000 000 vertices.



(a) Transfomed model with the cage and the scan from the front.

(b) Transfomed model with the scan from the front.

(c) Transfomed model with the scan from the side.

(d) Transfomed locking mechanism with the scan from the side.

Fig. 10. Transformed model of the cranial orthosis to specific size and shape of the head

7 Conclusion

We have provided suitable method for morphing of a complex 3D model of a cranial orthosis. Method is based on Radial Basis Functions (RBFs) and it can adjust the size and shape of the orthosis from the general one to the customized one based on the scan of the patient's head. The method can perform non-rigid transformation of the orthosis together with rigid transformation of its specific part that has to preserve the size and the shape for its proper functionality. Our method can therefore provide virtual designing process of the orthosis to specific patient's needs without the necessity to create any physical prototypes that would need to be manually adjusted. Our method has been also speeded up by OpenMP framework and it can efficiently utilize available resources of a computer.

Acknowledgement. This work was supported by The Ministry of Education, Youth and Sports from the Large Infrastructures for Research, Experimental Development and Innovations project "IT4Innovations National Supercomputing Center - LM2015070". This work is the main objective of contractual research conducted in collaboration with ING corporation spol. s.r.o.

References

1. Hardware Overview - IT4Innovations Documentation, August 2017. https://docs.it4i.cz/salomon/hardware-overview/
2. de Boer, A., van der Schoot, M., Bijl, H.: Mesh deformation based on radial basis function interpolation. Comput. Struct. **85**(11), 784–795 (2007). https://doi.org/10.1016/j.compstruc.2007.01.013. http://www.sciencedirect.com/science/article/pii/S0045794907000223. Fourth MIT Conference on Computational Fluid and Solid Mechanics
3. Clarren, S.K.: Plagiocephaly and torticollis: etiology, natural history, and helmet treatment. J. Pediatr. **98**(1), 92–95 (1981). http://www.sciencedirect.com/science/article/pii/S0022347681805495
4. Hotelling, H.: Analysis of a Complex of Statistical Variables Into Principal Components. Warwick & York (1933). https://books.google.cz/books?id=qJfXAAAAMAAJ
5. Joshi, P., Meyer, M., DeRose, T., Green, B., Sanocki, T.: Harmonic coordinates for character articulation. ACM Trans. Graph. **26**(3) (2007). https://doi.org/10.1145/1276377.1276466
6. Little, J., Hill, D., Hawkes, D.: Deformations incorporating rigid structures. Comput. Vis. Image Underst. **66**(2), 223–232 (1997). https://doi.org/10.1006/cviu.1997.0608. http://www.sciencedirect.com/science/article/pii/S1077314297906081
7. Modersitzki, J.: Numerical Methods for Image Registration. Oxford University Press, Oxford (2004)
8. Pearson, K.: On lines and planes of closest fit to systems of points in space. Philos. Mag. **2**(11), 559–572 (1901). https://doi.org/10.1080/14786440109462720
9. Persing, J., James, H., Swanson, J., Kattwinkel, J.: Prevention and management of positional skull deformities in infants. Pediatrics **112**(1), 199–202 (2003). http://pediatrics.aappublications.org/content/112/1/199

10. Schreen, G., Matarazzo, C.G.: Plagiocephaly and brachycephaly treatment with cranial orthosis: a case report. Einstein **11**, 114–118 (2013). http://www.scielo.br/scielo.php?script=sci_arttext&pid=S1679-45082013000100021&nrm=iso
11. Schroeder, W., Martin, K.M., Lorensen, W.E.: The Visualization Toolkit: An Object-oriented Approach to 3D Graphics, 2nd edn. Prentice-Hall Inc, Upper Saddle River (1998)
12. Sieger, D., Menzel, S., Botsch, M.: High quality mesh morphing using triharmonic radial basis functions. In: Jiao, X., Weill, J.C. (eds.) Proceedings of the 21st International Meshing Roundtable, pp. 1–15. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33573-0_1