Computers & Graphics 34 (2010) 378–387

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

Technical Section

Layout-aware optimization for interactive labeling of 3D models

Ladislav Čmolík *, Jiří Bittner

Faculty of Electrical Engineering, Czech Technical University in Prague, Czech Republic

ARTICLE INFO

Keywords:
Illustration
Labeling
Interaction

ABSTRACT

We propose a novel method for computing labeling of 3D illustrations in real-time. We solve a multiple criteria optimization problem in which we consider the desired layout already in the stage of searching for salient points of the labeled areas. In the solution we employ fuzzy logic combined with greedy optimization. The method runs on the GPU and achieves interactive rates on medium sized models. The results indicate that the method compares favorably to the state-of-the-art interactive labeling techniques.

1. Introduction

Illustrations are an important visual component of communication. They are used to visually expound various objects and support their textual description. In the latter case the reader needs to link the terms contained in the text with the illustration. The relation between textual and visual representations of information is mediated through *labeling*, i.e. assigning textual labels to various parts of the illustration. Digital media offer new possibilities for illustrations, such as 3D models, which the reader can manipulate interactively.

In this paper we present a novel labeling method which is targeted at interactive illustration of 3D models. The three main contributions of the paper are: (1) We formulate the labeling as multiple criteria optimization problem which considers the desired layout already in the stage of searching for salient points of labeled areas. This improves the resulting labeling compared to previous methods especially in the areas with many labels (see Fig. 1). (2) We use fuzzy logic and greedy optimization to solve the multiple criteria optimization problem. (3) We describe a GPU implementation of the method, which achieves interactive rates on medium sized models. Since the labeling is recomputed every frame, our method supports arbitrary manipulations of the model as well as interactive modifications of the model and of the labels.

The paper is organized as follows: Section 2 introduces terms used in the area of labeling. Section 3 summarizes state-of-the-art in the area of labeling. Section 4 formally describes the problem of external labeling. Section 5 presents our solution to the problem, which is summarized once more in Section 6. Section 7 presents results and comparisons and finally Section 8 concludes the paper.

2. The labeling problem

In this section we describe the labeling problem and define terminology used later in the paper.

2.1. Basic terminology

We assume that the model consists of n objects $O_i, 1 \leq i \leq n$ and each object O_i is assigned a unique label. After projection of the model to the screen, object O_i becomes visible in the screen area A_i. Note that if O_i is invisible then $A_i = \emptyset$.

The interior area A_I is a superset of the union of A_i over all objects. In our case we deal with a convex A_I, which is constructed to include a small boundary area around the model. The exterior area A_E is the complement of A_I with respect to the total screen area A_S ($A_E = A_S - A_I$). If the labels are placed in the interior area we call the labeling *internal*. If the labels are placed in the exterior area we call the labeling *external*. Our method deals with external labeling and thus we describe it in more detail in the next section.

2.2. External labeling

In external labeling a label is associated with the *anchor*, the *leader line*, and the *label box*. The anchor $\mathbf{a}_i$ is a point inside the area A_i. The label box L_i is a rectangle containing the label typically in the form of a short text string. Leader line $\mathbf{l}_i$ is a line segment connecting the anchor $\mathbf{a}_i$ and the label box. The endpoint of the leader line is denoted $\mathbf{e}_i$ (see Fig. 3).

The label boxes in external labeling can be either floating or fixed. A floating label box can be placed at any position in the external area while a fixed label box can be placed only at several fixed positions (the number of these positions is typically the same as the number of label boxes).

* Corresponding author.
E-mail addresses: cmolikl@fel.cvut.cz (L. Čmolík), bittner@fel.cvut.cz (J. Bittner).

0097-8493/$ - see front matter © 2010 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cag.2010.05.002

Similarly, the anchors and endpoints of leader lines can be floating or fixed. A floating anchor can be placed at any position inside the corresponding area while the fixed anchor has a one or several fixed positions. A floating endpoint can be placed at any position on the boundary of the label box while a fixed endpoint has only one or a few fixed positions. We call a labeling method *automatic* if it deals with both floating anchors and floating endpoints.

With floating label boxes, a set of *principal* directions D can be used to specify the desired layout of the leader lines. Then each leader line l_i should be parallel to some principal direction $d \in D$. However, this is not always possible for all leader lines without introducing overlaps of the leader lines or the label boxes. If this happens there are two commonly used solutions:

- The leader lines remain straight lines, but some of them are no longer parallel to any principal direction.
- Some of the leader lines are split into two orthogonal lines with one bend, where the segment from the anchor to the bend is orthogonal to $d \in D$ and the segment from the bend to the endpoint is parallel with d.

Examples of layouts with different sets of principal directions and type of leader lines are shown in Fig. 2.

3. Related work

The labeling problem has first received attention in the cartographic domain for assigning labels to static features. A comprehensive bibliography of these labeling techniques can was presented by Wolff and Strijk [19].

Although we deal with external labeling we identified several methods for internal labeling that are related to our work: The method of Götzelmann et al. [8] determines the positions of internal labels using a multiple criteria optimization. In the method of Ropinski et al. [15] the labels indicate the shape of the overlaid part of the 3D object. The method of Maass and Döllner [13] integrates the labels into a virtual reality environment.

In the case of external labeling we split the discussion of the related work into four parts according to the flexibility of anchors and label boxes (fixed vs. floating).

Fixed anchors and fixed label boxes: Bekos et al. [3] defined the boundary labeling problem where the label boxes are arranged on the rectangle enclosing a set of anchors. They study various types of leader lines, arrangements of label boxes and sizes of label boxes. Their primary focus is on efficient labeling algorithms that calculate leader lines whose combined length is minimal. Later, Benkert et al. [5] formulated the boundary labeling problem as a multiple criteria optimization problem where the length of leader lines, the number of bends, and the distance of anchors to leader lines are used to find an optimal solution of one-sided labeling (all label boxes are on one side of the enclosing rectangle).

Floating anchors and fixed label boxes: Bekos et al. [2] extended the boundary labeling problem. Each anchor can float within a polygonal area. They propose efficient labeling algorithms for various types of leader lines under some restrictions on the polygonal area with the aim of minimizing the combined length of leader lines.

Fixed anchors and floating label boxes: Stein and Décoret [17] presented a greedy algorithm for the labeling of fixed anchors attached to 3D objects. The occlusion of the 3D objects is minimized by placing label boxes in empty areas. Shadow regions and a summed area table [11] are used to prevent the crossing of leader lines and overlaps of label boxes.

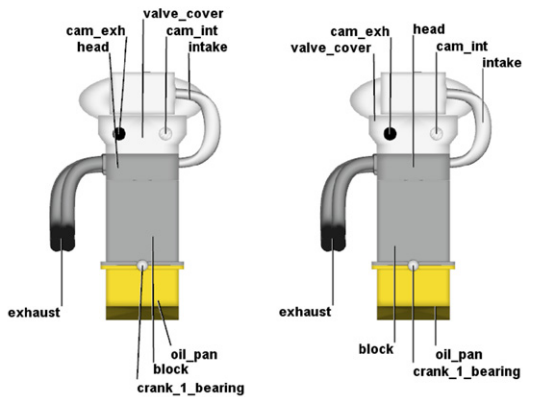

Fig. 1. A comparison of our method with the method of Ali et al. [1] on an engine model using top–bottom layout. (left) Ali et al. [1], (right) the proposed technique. Note that in our method the leader lines are distributed more evenly over the model, which according to our opinion increases their saliency and leads to more aesthetic labeling.

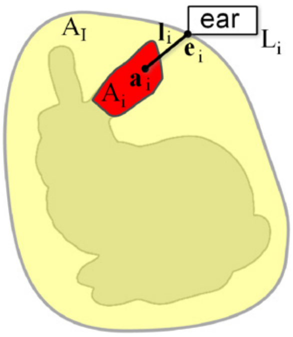

Fig. 3. Illustration of the basic terms used in external labeling.

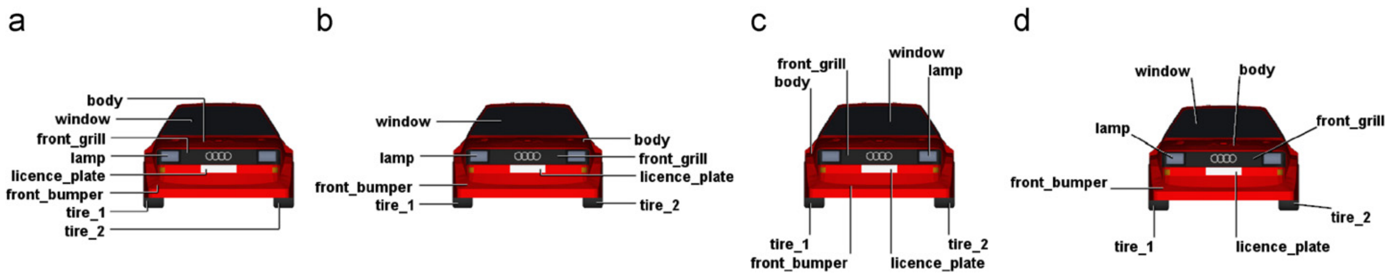

Fig. 2. Examples of different label layouts: (a) left layout with orthogonal lines, (b) left–right layout with orthogonal lines, (c) top–bottom layout with straight lines, (d) silhouette based layout with straight lines.

Floating anchors and floating label boxes: Hartmann et al. [9] introduced a method to determine the labeling of 2D and 3D objects based on dynamic potential fields. The problem is split into finding anchors for the objects and labeling those anchors. The labeling is obtained as an equilibrium between attractive and compulsive forces established for the label boxes and the objects. Ali et al. [1] presented a real-time labeling pipeline, allowing to produce various labeling styles of 3D objects. The problem is again split into finding anchors of 3D objects and labeling those anchors. This method is able to calculate the labeling of 3D models with frame-to-frame coherence at interactive frame rates. Götzelmann et al. [6] presented an agent-based labeling system, allowing the integration of internal and external labels. Also here the problem is split into finding anchors for 3D objects and labeling those anchors. Agents are assigned to initial labels and they compete and/or cooperate to meet metrics for functional and aesthetic label layouts [10], extracted from handmade illustrations.

There are also methods that do not fit into the classification according to anchor and label box properties, such as the method of Götzelmann et al. [7] for labeling animated objects or the method of Vollick et al. [18], which is able to learn a specific labeling style from given examples and to apply the style to new illustrations.

4. External labeling as optimization problem

In this section we review the criteria for finding a good external labeling.

4.1. Criteria for external labeling

A labeling of an illustration should exhibit four general properties: *readability*, *unambiguity*, *aesthetics*, and *compactness*. In order to allow an automatic computation of the labeling these properties are usually transformed into a number of criteria, which deal with the positions of the leader lines, label boxes, and anchors [9,10]:

(I) *Leader line crossing*: Leader lines do not cross.
(II) *Leader line distance*: Leader lines are not too near to each other.
(III) *Leader line alignment*: Leader lines are aligned with principal directions.
(IV) *Leader line length*: Leader lines are as short as possible.
(V) *Label box distance*: Label boxes are near to the corresponding objects.
(VI) *Label box overlap*: Label boxes do not overlap.
(VII) *Anchor salience*: Anchors are salient points of the corresponding areas.
(VIII) *Anchor distance*: Anchors $\mathbf{a}_i$ are not too near to each other.
(IX) *Endpoint distance*: Endpoints $\mathbf{e}_i$ are not too near to each other.
(X) *Label box coherence*: Label boxes in frame t are close to their positions in frame $t-1$.
(XI) *Anchor coherence*: Anchors in frame t are close to their positions in frame $t-1$.

Note that the last two criteria are important for interactive applications, where the labeling of the model should exhibit temporal coherence, i.e., the leader lines and the labels should not jump from one frame to the next. Note that in static applications these two criteria can be neglected. Some of the criteria contradict each other (e.g., criterion (IV) and criterion (VII)) and thus we have to find a balance between the contradicting criteria. Finding the importance of the contradicting criteria is an issue dealing with human perception and in general there is no unique optimal solution to the external labeling problem.

5. Computing optimized external labeling

If we analyze the methods suitable for automatic interactive external labeling of 3D objects [1,6], we observe that these methods proceed in three steps: (1) calculate the anchors, (2) calculate the initial layout for the anchors, (3) correct the initial layout. This approach, however, has one drawback: As the calculation of anchor points in the first step is not using any information about the layout, the distribution of these points may lead to situations in which the labeling computed in the second step contains clusters of overlapping labels. In the third step these clusters can be resolved by repositioning the labels, but in general this leads to undesired bends or rotations of the leader lines. In our approach, we improve these approaches by combining the optimization of anchor determination with optimizing the layout of the labels.

5.1. Overview of our approach

We use a greedy optimization to determine the positions of the anchors, the leader lines, and the label boxes. As a leader line connects the anchor with a label box, we only deal with finding a good set of leader lines. Leader lines are calculated and placed over the illustration by repeating the following two steps:

1. Select an unlabeled area.
2. Calculate the leader line of the area.

In the first step we select an area which has not been labeled so far. We process first the areas where there is less freedom for placing the labels (i.e., small areas in dense regions of the model). The area selection is described in detail in Section 5.4. The second step forms the core of the method. We search for a good leader line considering all feasible anchor points and principal directions. The leader line computation is described in Section 5.3. In the next section we discuss the criteria which we later use for both the area selection and the leader line computation.

5.2. Simplifying the criteria for optimization

Our method performs the determination of the anchors and the calculation of the label layout together, which allows to improve the results in some difficult situations with a high density of labels. However, optimizing both the positions of the anchors and the label boxes is more complicated than optimizing the position of just one of these features as done by previous interactive methods.

In particular when placing a single floating leader line over the illustration we have four degrees of freedom (two for the anchor point and two for the end point) and so we deal with a 4D problem. However, we can reduce the dimensionality of the problem by making use of the labeling criteria dealing with principal directions by defining a unique leader line with respect to each anchor point as follows: According to the above listed criteria (III) and (IV) this line is determined as the shortest line with respect to the silhouette of A_I which follows a principal direction. More precisely, for a leader line $\mathbf{l}_i = (\mathbf{a}_i, \mathbf{e}_i)$, $\mathbf{e}_i$ is the point on the silhouette of A_I closest to $\mathbf{a}_i$ (criterion (IV)) such that $\mathbf{a}_i - \mathbf{e}_i$ is parallel to a principal direction (criterion (III)). In this way we reduce the number of possible directions that the leader line can follow to a single line (see Fig. 4(a)). In turn we reduce the dimensionality of the problem from 4D to 2D which allows to us to represent the measures associated with the quality of every leader line candidate directly in the labeled image.

Following this mapping we can simplify the criteria for labeling as follows: The leader lines defined using the described mapping will never cross (assuming A_I is convex), therefore the criterion (I) (leader line crossing) can be omitted.

Criterion (II) (leader line distance) can be replaced by considering the distance between the anchors and between the endpoints of the leader lines.

We can also replace the criteria for the label boxes by criteria for the endpoints and anchors. If leader lines are short (criterion (IV)) then each label box is near to its corresponding object. Therefore the criterion (V) can be omitted. The criterion (IV) is replaced as follows: The leader lines do not cross and they are not even near each other if the distance between each pair of endpoints $\mathbf{e}_i,\mathbf{e}_j, i \neq j$, is greater than a threshold $t > 0$. If the threshold t is greater than the height of the label box and the internal area A_I is convex, then the label boxes can be stacked around silhouette of A_I without overlap. See Fig. 4(b) for details.

In summary, by using the above described mapping and a convex shape for describing the internal area A_I we can simplify the labeling criteria to the following seven criteria (we use Arabic numbers in order to distinguish between the original and the simplified criteria):

1. *Leader line alignment*: Leader lines are aligned with principal directions.

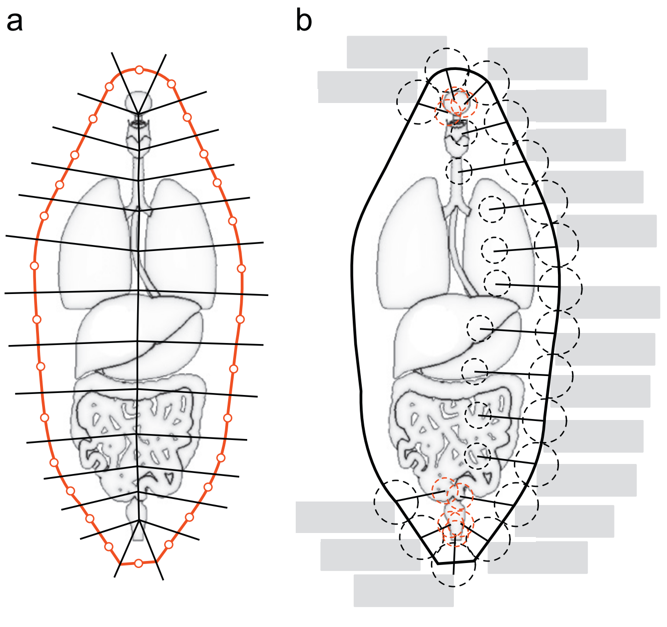

Fig. 4. (a) Voronoi diagram of points on the silhouette of A_I. Note that as we add more points on the silhouette the cells become thinner. When we use all points on the silhouette the cells will collapse into lines and thus we get a leader line candidate for each point in A_I. (b) Stacking of label boxes proposed by Ali et al. [1]. Label boxes can be stacked around the silhouette of a convex A_I if there is space around each leader line endpoint equal or bigger than the height of the label box. Note that in the upper and lower parts some leader lines have to be extruded (e.g., the leader line of the lowest label box). If we demand a larger distance between the anchors we have to choose only one leader line from the two overlapping ones (marked by a red circle around the anchor). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

2. *Leader line length*: Leader lines are as short as possible.
3. *Anchor salience*: Anchors are salient points of the corresponding areas.
4. *Anchor distance*: Anchors $\mathbf{a}_i$ are not too near to each other.
5. *Endpoint distance*: Endpoints $\mathbf{e}_i$ are not too near to each other.
6. *Label box coherence*: Label boxes in frame t are close to their positions in frame $t-1$.
7. *Anchor coherence*: Anchors in frame t are close to their positions in frame $t-1$.

These seven criteria are used in the optimization framework for finding the most suitable labeling as described in the following sections.

5.3. Leader line calculation

Given an area A_i we need to find a leader line which follows the criteria listed in the previous section. For every point in A_i we have a unique *leader line candidate*. From these candidates the algorithm selects the leader line which provides the best match for the labeling criteria.

In order to do so we use fuzzy optimization [21] based on fuzzy decision making by Bellman and Zadeh [4]. A process where simultaneous satisfaction of several criteria is sought.

We see the following benefits in using the fuzzy optimization. It can handle conflicting criteria due to the simultaneous satisfaction of all criteria. It can handle uncertainty in the criteria due to its basis in the fuzzy set theory and fuzzy logic.

Fuzzy set theory is an extension of set theory. In fuzzy set theory we can express a partial membership of an element to the set. A fuzzy set is commonly described by its membership function that maps each element to values in the range [0,1] which indicates the membership of the element to the set, 0 means that the element is not in the set and 1 means that the element is entirely in the set. Fuzzy logic defines operations on the fuzzy sets that are equivalents of Boolean logic operations (e.g., negation, conjunction, and disjunction).

In the fuzzy optimization we consider a solution space X. Each criterion C_i is modeled as a fuzzy set on X and its membership function f_i describes the satisfaction of the criterion by the solutions $x \in X$. The membership functions are aggregated together using an intersection operator (e.g., fuzzy conjunction), thus we obtain the aggregated membership function f for the criteria:

$$f(x) = \bigcap_{1 \le i \le 7} f_i(x). \tag{1}$$

We denote the value returned by $f(x)$ as feasibility of solution x. Note that the intersection operator guarantees the simultaneous satisfaction of the criteria. In other words, satisfaction of one criterion cannot compensate dissatisfaction of another criterion. The last step is to find the most feasible solution, that is the global maximum of the aggregated membership function $f(x)$.

In the following, we model the satisfaction of our labeling criteria as fuzzy sets and define their membership functions. In our case the solution space are the leader line candidates of area A_i for which we search the leader line, which provides best match for all criteria. The criterion 1 is satisfied implicitly since we consider only leader line candidates which are aligned with principal directions. The fuzzy membership functions for a leader line candidate $\mathbf{l} = (\mathbf{a}, \mathbf{e})$ of area A_i for criteria 2–7 are evaluated as follows:

Leader line length: To express the leader line length we calculate the distance of the anchor from the endpoint and normalize it using the length of the longest leader line candidate:

$$f_2(\mathbf{l}) = 1 - \frac{|\mathbf{a}-\mathbf{e}|}{d_{max}}, \tag{2}$$

where $|\mathbf{a}-\mathbf{e}|$ is the distance of points $\mathbf{a}$ and $\mathbf{e}$ and d_{max} is the length of the longest leader line candidate (see Fig. 5(b)).

Anchor salience: To express the anchor salience we compute the distance of the anchor from the silhouette of A_i and normalize it using the length of the longest leader line candidate:

$$f_3(\mathbf{l}) = \frac{dist_{sil}(\mathbf{a})}{d_{max}}, \tag{3}$$

where $dist_{sil}$ is a distance of the anchor to the silhouette of area A_i and d_{max} is length of the longest leader line candidate (see Fig. 5(c)).

Anchor distance: To express the distance from other anchors we use the minimal distance of the anchor to already placed leader lines $\mathbf{p} \in P$. The distance is normalized using the threshold d_a, which reflects the desired minimal distance between the anchors. For a given leader line candidate and an already placed leader line $\mathbf{p}$ we get

$$dist_a(\mathbf{l},\mathbf{p}) = \min\left(\frac{|\mathbf{a}-\mathbf{a_p}|}{d_a}, 1\right), \tag{4}$$

where $|\mathbf{a}-\mathbf{a_p}|$ is the distance of the anchors. To compute membership function f_4 with respect to all already placed leader lines P we use a conjunction of $dist_a$:

$$f_4(\mathbf{l}) = \bigwedge_{\mathbf{p} \in P} dist_a(\mathbf{l},\mathbf{p}). \tag{5}$$

Fig. 5(d) shows a visualization of this membership function.

Endpoint distance: To express the distance from the endpoints we use the minimal distance of the endpoint of the leader line to endpoints of already placed leader lines $\mathbf{p} \in P$. The distance is normalized using the threshold d_e, which reflects the desired minimal distance between the anchors. For a given leader line candidate and an already placed leader line $\mathbf{p}$ we get

$$dist_e(\mathbf{l},\mathbf{p}) = \min\left(\frac{|\mathbf{e}-\mathbf{e_p}|}{d_e}, 1\right). \tag{6}$$

To compute membership function f_5 with respect to all already placed leader lines P we use a conjunction of $dist_e$:

$$f_5(\mathbf{l}) = \bigwedge_{\mathbf{p} \in P} dist_e(\mathbf{l},\mathbf{p}). \tag{7}$$

Fig. 5(e) shows a visualization of this membership function.

Anchor coherence: The anchor coherence is expressed by calculating the distance of the corresponding anchors in two consecutive frames and normalizing it using the diagonal of the bounding sphere d_S:

$$f_6(\mathbf{l}) = 1 - \frac{|\mathbf{a}_t - \mathbf{a}_{t-1}|}{d_S}, \tag{8}$$

where $|\mathbf{a}_t - \mathbf{a}_{t-1}|$ is the distance of the corresponding anchor points in frames t and $t-1$.

Endpoint coherence: To express the endpoint coherence we calculate the distance of the corresponding endpoints in two consecutive frames and normalize it using the diagonal of the bounding sphere d_S:

$$f_7(\mathbf{l}) = 1 - \frac{|\mathbf{e}_t - \mathbf{e}_{t-1}|}{d_S}, \tag{9}$$

where $|\mathbf{e}_t - \mathbf{e}_{t-1}|$ is the distance of the corresponding endpoints in frames t and $t-1$.

To obtain the aggregated membership function of all criteria we use the natural fuzzy conjunction [21] as the intersection operator in Eq. (1). Thus, we get

$$f(\mathbf{l}) = \bigwedge_{2 \le i \le 7} f_i(\mathbf{l}). \tag{10}$$

Natural fuzzy conjunction corresponds to a simple multiplication. Note that the use of multiplication is robust with respect to scaling one, several, or all membership functions, i.e., the most feasible solution does not change.

If we want to tune the behavior of the labeling we can define a weight for each criterion. Here we use the weighted modification

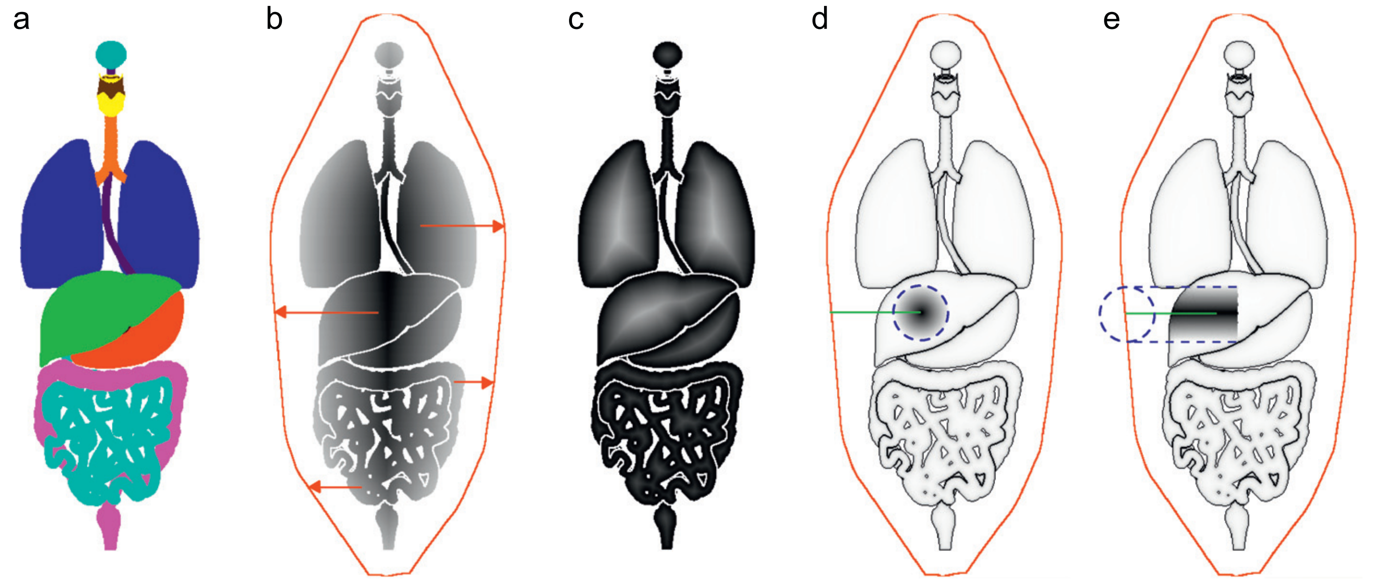

Fig. 5. (a) Colour coded areas A_i. (b–e) Visualizations of membership functions: (b) f_2—leader line length, (c) f_3—anchor salience, (d) f_4—anchor distance, (e) f_5—endpoint distance. A darker pixel corresponds to a less feasible leader line candidate. We show the membership functions for left–right layout and therefore only the distances to the convex hull in left and right direction are considered in figure (b). Figures (d) and (e) show the corresponding membership functions after one leader line has been placed over the illustration (depicted in green colour). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

of fuzzy optimization introduced by Yager [20], who proposed to modify the membership function of each criterion prior to the aggregation. The modified function F_i is expressed as

$$F_i(\mathbf{l},w_i)=f_i(\mathbf{l})^{w_i}, \tag{11}$$

where f_i is the membership function and w_i is the weight in range [0, 1]. Eq. (1) is modified to

$$F(\mathbf{l})=\bigwedge_{2\le i\le 7} F_i(\mathbf{l},w_i). \tag{12}$$

Note that if $f_i'(\mathbf{l})=c\cdot f_i(\mathbf{l})$ where c is a constant, then $F_i'(\mathbf{l},w_i)=f_i'(\mathbf{l})^{w_i}=c^{w_i}\cdot f_i(\mathbf{l})^{w_i}=c^{w_i}\cdot F_i(\mathbf{l},w_i)$ where c^{w_i} is also a constant. Therefore the weighted fuzzy optimization is resistant to scaling.

Note that all described functions are evaluated on a discrete grid corresponding to the image pixels (see Fig. 5). In the evaluation we make heavy use of the jump flooding algorithm for computing distance fields. We provide more details about the implementation in Section 6.2.

5.4. Processing order of labeled areas

We use a greedy optimization technique without backtracking and therefore the quality of the final labeling depends on the order in which the leader lines are placed over the illustration. Each leader line placed over the illustration potentially reduces the possibilities for placing leader lines for other areas.

We use the following strategy for determining the processing order of the labeled areas: we first process the areas that could be heavily influenced by leader lines of other areas. The presumption is that the area with a low number of feasible leader line candidates can get influenced much more than an area with high number of feasible candidates and therefore it should be processed first.

Thus for each area we sum the feasibility of all corresponding leader lines obtained with Eq. (12) and define the priority p_i of the area A_i as

$$p_i=\sum_{\mathbf{l}\in A_i}(1-f(\mathbf{l})). \tag{13}$$

For the weighted conjunction we get

$$P_i=\sum_{\mathbf{l}\in A_i}(1-F(\mathbf{l})). \tag{14}$$

Initially we evaluate priorities for all areas and select the area with the highest priority as the next area for placing a label. After the leader line is calculated we reevaluate the priorities of all unlabeled areas and proceed again by selecting the area with the highest priority.

5.5. Corrections of label layout

Note that in some cases not all labeling criteria can be satisfied at the same time. Especially the satisfaction of the criterion 5 (endpoint distance) is crucial, which, if violated, would result in overlapping label boxes. Fortunately, this can happen only if there is no space left to place new leader lines and the violation can be easily detected and corrected by repositioning the label boxes to avoid overlaps and relieving the criterion 1, i.e., splitting or rotating the leader lines [1]. This in turn may cause intersections of leader lines (the criterion 1 is waived). If this happens, the intersections have to be detected and resolved by swapping label box positions [1,2].

5.6. Defining the layout types

The layout of the label boxes is determined by two factors: the shape of the internal area A_I, and the set of principal directions D. A feasible leader line is the shortest line from an anchor to the silhouette of internal area A_I that is parallel with one of principal directions $\mathbf{d}\in D$. The shape of the internal area A_I then determines the length of leader line in the possible directions and the shortest one is chosen. In case that the set of principal directions is not restricted the leader line is simply the shortest line from the anchor to the silhouette of A_I.

If all directions are used as principal directions then both the directions of leader lines and the positions of the label boxes are only given by the shape of the internal area A_I. If only certain directions are used as principal directions then the directions of leader lines are given by the principal directions and the positions of label boxes are given by the shape of the internal area A_I.

6. Putting it together

In this section we briefly review the complete algorithm and describe some implementation details, particularly those connected with the GPU implementation of the method.

Table 1
Look up directions used in the jump flooding algorithm for different layout types.

Layout type	Principal directions	Jump flooding look up directions
Left	West	West
Right	East	East
Left-right	West, East	West, East
Top	North	North
Bottom	South	South
Top–bottom	North, South	North, South
Silhouette-based	All directions	West, East, North, South, Northwest, Northeast, Southwest, Southeast

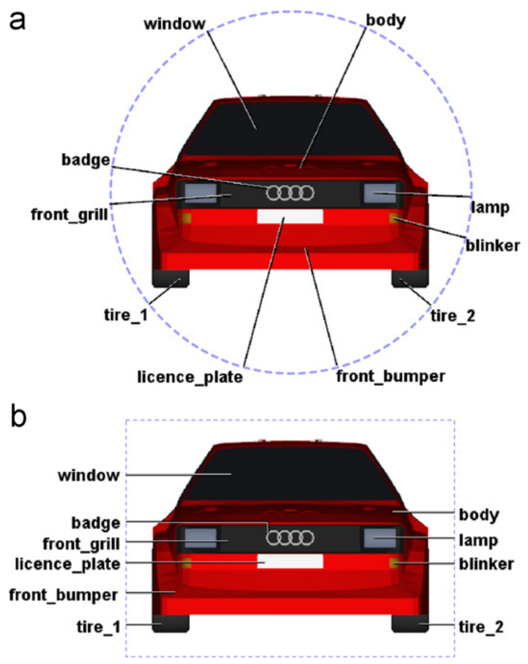

Fig. 6. Label layouts obtained by using: (a) a circle enclosing the 3D model, (b) a rectangle enclosing the 3D model instead of convex hull of the 3D model.

6.1. Summary of the algorithm

The proposed algorithm takes as input a 3D scene S consisting of n 3D objects $O_i, i \in \{1,\ldots,n\}$, the set of principal directions D, the parameters d_a and d_e, and the weights $w_1, \ldots, w_7$. The result of the algorithm is a set of leader lines $\mathbb{L} = \{\mathbf{l}_1 = (\mathbf{a}_1,\mathbf{e}_1),\ldots,\mathbf{l}_n = (\mathbf{a}_n,\mathbf{e}_n)\}$. The algorithm proceeds as follows:

1. Obtain the set of areas $\mathbb{A} = \{A_1,\ldots,A_n\}$ by computing the visibility of each object in scene S.
2. For each point $\mathbf{a}$ in internal area A_I find the closest point $\mathbf{e}$ on the silhouette of A_I such that $\mathbf{a} - \mathbf{e}$ is collinear with a direction $\mathbf{d} \in D$. Thus the leader line candidate $\mathbf{l} = (\mathbf{a}, \mathbf{e})$ is established and function $f_2(\mathbf{l})$ is calculated for each point $\mathbf{a}$ in A_I.
3. Calculate the length of the longest leader line candidate d_{max}.
4. For each point $\mathbf{a}$ in area $A_i, i \in \{1,\ldots,n\}$, calculate the function $f_3(\mathbf{l})$. That is, the distance of $\mathbf{a}$ to the nearest point on the silhouette of A_i.
5. Establish the set of leader lines $\mathbb{L} = \{\}$.
6. Calculate the feasibility $F(\mathbf{l})$ of each leader line candidate $\mathbf{l} = (\mathbf{a}, \mathbf{e})$.
7. While the set of areas is not empty, do
   (a) Calculate the priority P_i for each area A_i.
   (b) Select the area A_{max} with the highest priority.
   (c) Select the most feasible leader line candidate $\mathbf{l}_{max}$ with maximal $F(\mathbf{l})$.
   (d) Put the leader line candidate $\mathbf{l}_{max}$ into the set of leader lines $\mathbb{L}$.
   (e) Remove the area A_{max} from the set of areas $\mathbb{A}$.
   (f) Update functions f_3 and f_4 using $\mathbf{l}_{max}$ and reevaluate $F(\mathbf{l})$ for all leader line candidates in non-processed areas $\mathbb{A}$.
8. If necessary correct the label layout.

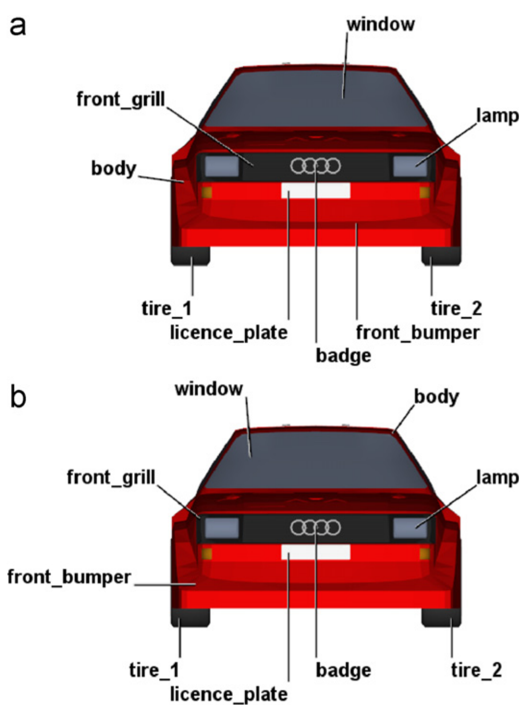

Fig. 7. The influence of the parameter d_a on the final labeling: (a) $d_a = 0.1$, (b) $d_a = 0.4$. We used $d_e = 0$ for both figures.

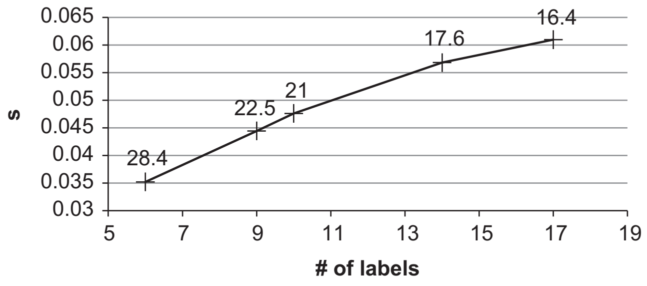

Fig. 8. The average time per frame in dependency on the number of labels. The number above each sample is the corresponding FPS rate.

6.2. Implementation details

We have implemented the presented algorithm as multi-pass rendering algorithm written in Java using OpenGL and JOGL [12].

Step 2 is calculated with jump flooding [14]. Jump flooding is a fast method, suitable for the GPU, for obtaining a Voronoi diagram of the seeds and the Euclidean distance of each pixel to the nearest seed. The pixels on the silhouette of A_I are used as the seeds. Note that for each pixel we need to find the closest point along one of the principal directions on the silhouette of A_I. Traditional jump flooding computes the Euclidean distance along all possible directions which is suitable for the silhouette-based layout only. For layouts with a restricted set of principal directions (e.g., left–right layout) we have modified the jump flooding algorithm to compute the Euclidean distance only along these directions. In Table 1 we show side-by-side the principal directions for major layout types and the corresponding look up directions used to evaluate minimal distances in the jump flooding algorithm. Note that apart from the metric used in distance evaluation the resulting layout can also be modified by changing the shape of the internal area A_I (as discussed

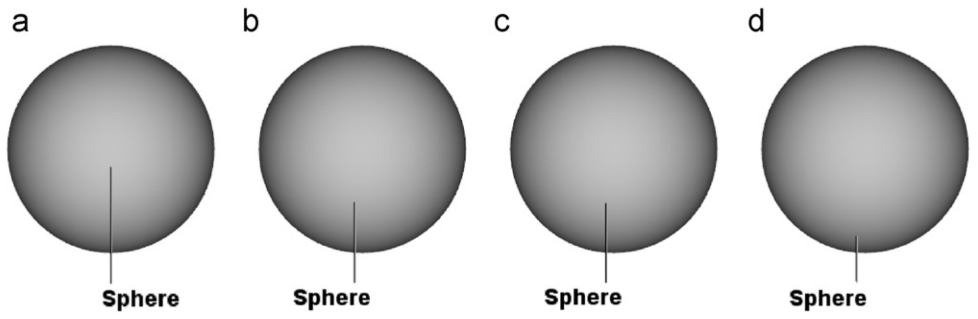

Fig. 9. A simple example showing the influence of the weights w_2 and w_3 on the labeling: (a) $w_2 = 0.2, w_3 = 1.0$, (b) $w_2 = 1.0, w_3 = 1.0$, (c) $w_2 = 0.2, w_3 = 0.2$, (d) $w_2 = 1.0$, $w_3 = 0.2$.

in Sections 5.6 and 7.1). An important aspect of jump flooding is that the texture in which we compute the distance has to be rectangular. Note that for representing the distances, the membership functions and their aggregations we use 16 bit floating point textures.

In step 3 we use scattering [16] to find the longest leader line candidate among all areas.

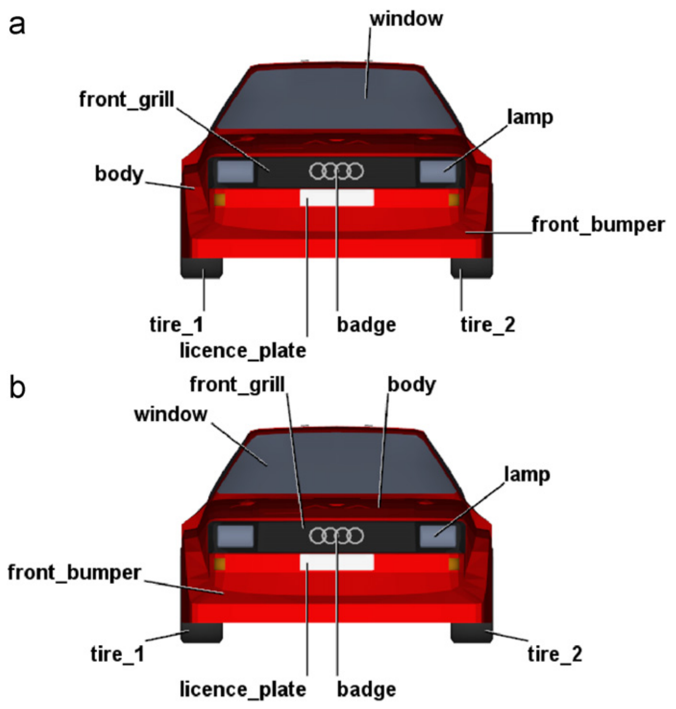

Fig. 10. The influence of the parameter d_e on the final labeling: (a) $d_e = 0.1$, (b) $d_e = 0.4$. We used $d_a = 0$ for both figures.

Step 4 is also calculated with jump flooding, where the seeds are pixels on the silhouettes of areas $A_1, \ldots, A_n$. For each pixel in area A_i, $i \in \{1, \ldots, n\}$, we calculate the Euclidean distance (along all possible directions) to the closest pixel on the silhouette of A_i. Note that the areas do not overlap each other, therefore we can process all areas at once.

Step 6 is implemented as a fragment shader working on a screen aligned quad. In the shader we calculate the feasibility $F(l)$ of each leader line candidate using Eq. (12).

In steps (a)–(c) we use scattering to compute the priorities P_i, selecting the area with the highest priority and selecting the most feasible leader line candidate.

In step 7(f) the functions $f_4(\mathbf{l})$ and $f_5(\mathbf{l})$ are calculated and multiplied with $F(\mathbf{l})$ using again a fragment shader operating on the screen aligned quad.

7. Results

In this section we present results of our solution of the external labeling problem and discuss the impact of the input parameters on the results. Further we compare our results with the method of Ali et al. [1].

7.1. Supported layout types

In order to demonstrate the flexibility of the labeling layouts produced by our method, we used our method with different shapes of internal area A_I and different sets of principal directions. When we use a convex hull of the 3D scene as internal area A_I and consider the distances only in certain principal directions we can obtain layouts presented in Figs. 1, 2, 7, 10, 11 and 12. When we use other shapes of the internal area we can obtain layouts as those presented in Fig. 6. In Fig. 6(a) principal directions correspond to all possible directions. In Fig. 6(b) only the east and west directions are used.

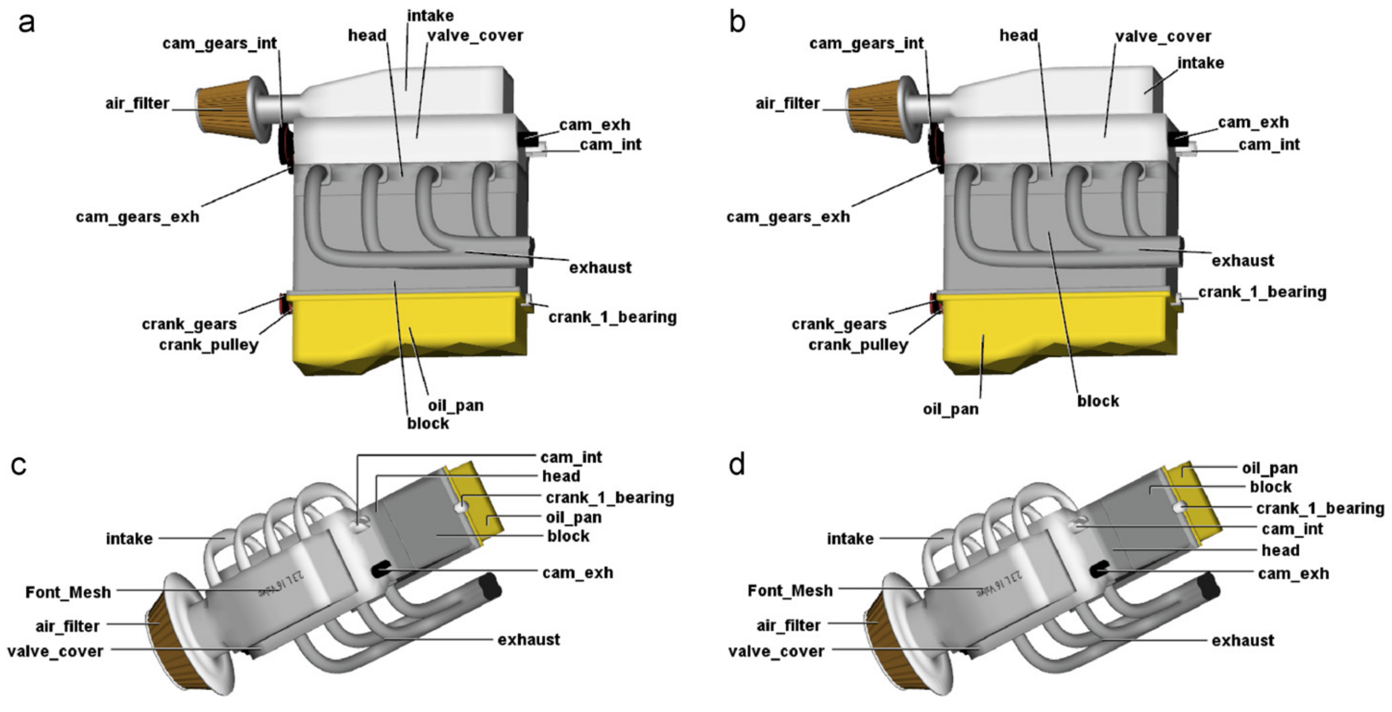

Fig. 11. A comparison of our method with the method of Ali et al. [1] on an engine model: (a) a silhouette-based layout produced using the method of Ali et al. ($d_a = 0.05$), (b) a silhouette-based layout using our method ($d_a = 0.05$, $d_e = 0.15$), (c) a left–right layout using the method of Ali et al. ($d_a = 0.15$), (d) a left–right layout using our method ($d_a = 0.15$, $d_e = 0.06$).

7.2. Distance of leader lines and distance of anchors

The input parameter d_a is used to specify the desired minimal distance between anchors. We express the distance as a scale of the bounding sphere diameter d_S. The impact of parameter d_a on the final labeling can be seen in Fig. 7.

The input parameter d_e is used to specify the desired minimal distance between the leader lines. Also here we express the distance as a scale of the bounding sphere diameter d_S. Our experiments have shown that superior values of d_e for horizontal layouts (left, right, and left–right) are similar to the height of the label box. For vertical layouts (top, bottom, and top–bottom) and radial layouts (silhouette-based) the superior values are similar to the average width of the label boxes. The impact of parameter d_e on the final labeling can be seen in Fig. 10.

Note that with increasing parameter d_a we can produce labeling where labels are more evenly distributed. However, certain anchors may become less salient (e.g. anchor to which label *body* is attached in Fig. 7(b)). When we also increase parameter d_e then the anchors are more resistant to this effect.

7.3. Influence of the weights

In this section we discus the impact of the weights on the final labeling. The weights are used to reduce the impact of the membership functions. In other words we use the weights to suppress significance of certain criteria and emphasize significance of the other criteria. As we use the same criteria for each type of layout we also use the same weights for all layout types. Note that we cannot change the type of layout by using different weights. We demonstrate the impact of the weights on a simple example in Fig. 9. In the example we use only criterion 1 (Leader line length) and criterion 2 (Anchor salience) and their respective weights.

The weights w_2 and w_3 influence the length of the leader lines and the salience of the anchors. The corresponding two criteria contradict each other and therefore a balance between them has to be found. Our experiments have shown that good values for w_2 and w_3 are: $w_2 = 0.2$ and $w_3 = 1$. In other words we prefer the salience of anchors over the length of the leader lines.

The weights w_4 and w_5 influence the distance between the anchors and the distance between the endpoints of leader lines. A similar effect can be achieved by using the parameters d_a and d_e.

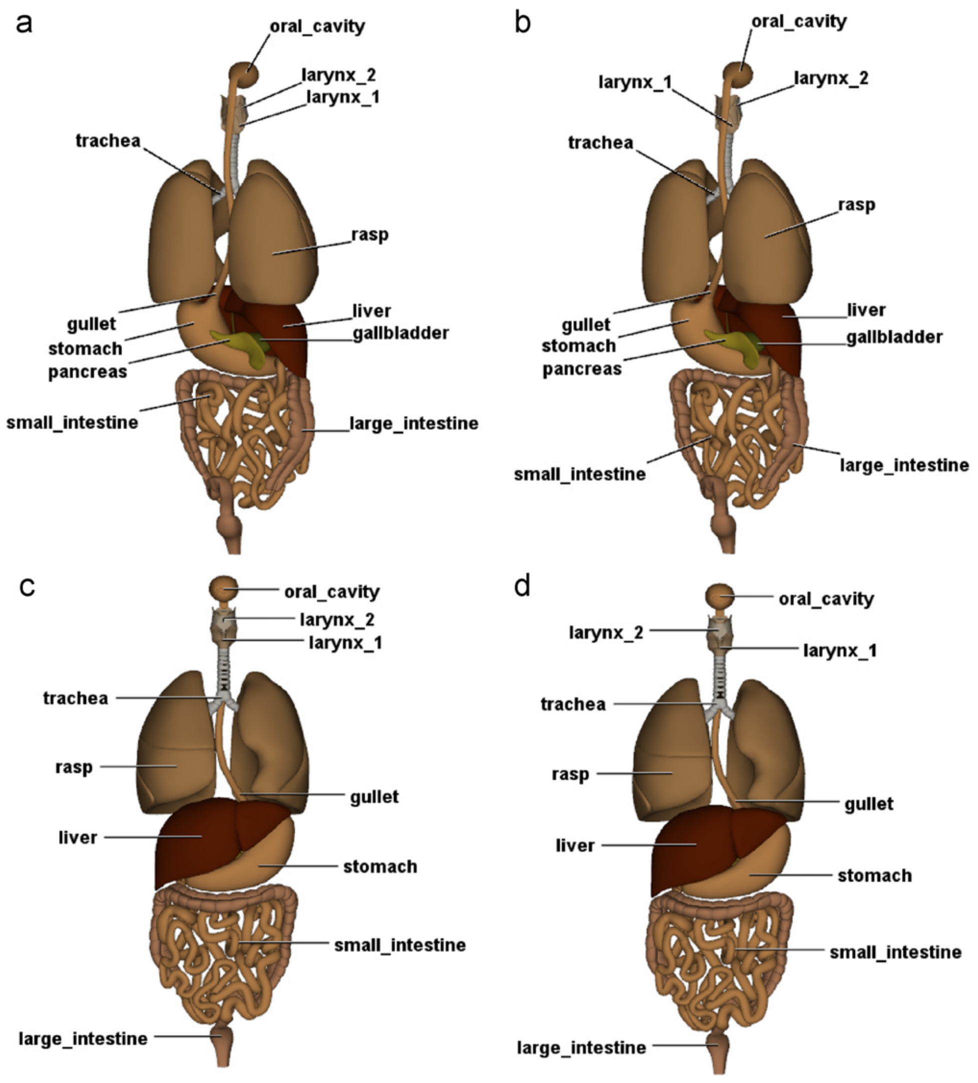

Fig. 12. A comparison of our method with the method of Ali et al. [1] on an anatomy model: (a) a silhouette-based layout produced using the method of Ali et al. ($d_a = 0.05$), (b) a silhouette-based layout using our method ($d_a = 0.05, d_e = 0.2$), (c) a left–right layout using the method of Ali et al. ($d_a = 0.05$), (d) a left–right layout using our method ($d_a = 0.05, d_e = 0.15$). Note particularly the difference in placing the larnix_1 and larnix_2 labels. Due to the endpoint distance criterion used in our method these labels are more uniformly distributed around the model.

These parameters provide more intuitive control over the labeling and therefore we recommend to set the weight w_4 and w_5 to 1 and use d_a and d_e to control the distance between the leader lines.

The weights w_6 and w_7 are used to reduce the impact of membership functions f_6 and f_7 that provide frame-to-frame coherence in interactive environments. Our experiments have shown that good values of w_6 and w_7 are: $w_6 = 0.7$ and $w_7 = 0.7$. These values appeared as best compromise between too glued anchors and too unstable leader lines.

Note that all figures in this paper obtained by our method were generated using weights with the above described values.

7.4. Comparison with state-of-the-art

In this section we compare the proposed method with our implementation of the method of Ali et al. [1]. A side by side comparison is shown in Figs. 11 and 12. As the method of Ali et al. does not consider criteria applying to the length of leader lines, the endpoint distance and the endpoint coherence it gives less aesthetic results for views where the most salient points of several areas are collinear. Note that for silhouette-based layout Ali et al. apply compulsive forces between the label boxes to reduce their uneven distribution. This step is omitted in our implementation of both their method and our method.

7.5. Performance

Fig. 8 shows the performance of our method in dependency on the number of labels. We can see that our method scales almost linearly with the number of labels. The measurements were done on a computer equipped with NVIDIA Geforce 8800 GT with 512 MB of RAM in resolution 512×512 pixels. Note that the number of labels depends on the number of visible objects. Typically the 3D model contains many more objects and only some of them are visible from a given viewpoint.

8. Conclusion

In this paper we have proposed a novel method for computing the labeling of 3D illustrations in real-time. We have shown how to reduce the dimensionality of the problem and how to formulate the labeling problem as a multiple criteria optimization problem. We solve the optimization problem using fuzzy logic combined with greedy optimization. We have implemented the presented method almost entirely on the GPU and the resulting implementation achieves interactive rates on medium-sized models. The results show that the method compares favorably to the state-of-the-art techniques for interactive external labeling. In particular, according to our opinion the method provides aesthetic labeling for various layout types and it is easy to fine tune the labeling by using a few intuitive parameters.

Acknowledgements

This research has been supported by the MSMT under the research programs LC-06008 (Center for Computer Graphics) and MEB060906 (Kontakt OE/CZ).

Appendix A. Supplementary material

Supplementary data associated with this article can be found in the online version of 10.1016/j.cag.2010.05.002.

References

[1] Ali K, Hartmann K, Strothotte T. Label layout for interactive 3D illustrations. Journal of the WSCG 2005;13(1):1–8.
[2] Bekos MA, Kaufmann M, Potika K, Symvonis A. Polygon labelling of minimum leader length. In: APVis '06: proceedings of the 2006 Asia-Pacific symposium on information visualisation. Darlinghurst, Australia: Australian Computer Society, Inc.; 2006. p. 15–21.
[3] Bekos MA, Kaufmann M, Symvonis A, Wolff A. Boundary labeling: models and efficient algorithms for rectangular maps. Computational Geometry 2007;36(3):215–36.
[4] Bellman RE, Zadeh LA. Decision-making in a fuzzy environment. Management Science 1970;17(4):B-141–64.
[5] Benkert M, Haverkort H, Kroll M, Nöllenburg M. Algorithms for multi-criteria one-sided boundary labeling. In: 15th international symposium on graph drawing, vol. 4875/2008. Berlin, Germany: Springer; 2008. p. 243–54.
[6] Götzelmann T, Hartmann K, Strothotte T. Agent-based annotation of interactive 3D visualizations. In: 6th international symposium on smart graphics, vol. 4073/2006. Berlin, Germany: Springer; 2006. p. 24–35.
[7] Götzelmann T, Hartmann K, Strothotte T. Annotation of animated 3D objects. In: Simulation und Visualisierung 2007 (SimVis 2007), Erlangen, Germany: SCS Publishing House; 2007. p. 209–22.
[8] Götzelmann T, Ali K, Hartmann K, Strothotte T. Form follows function: aesthetic interactive labels. In: Computational aesthetics 2005: eurographics workshop on computational aesthetics in graphics, visualization and imaging. Natick, MA, USA: A K Peters; 2005. p. 193–200.
[9] Hartmann K, Ali K, Strothotte T. Floating labels: applying dynamic potential fields for label layout. In: 4th international symposium on smart graphics, vol. 3031/2004. Berlin, Germany: Springer; 2004. p. 101–13.
[10] Hartmann K, Götzelmann T, Ali K, Strothotte T. Metrics for functional and aesthetic label layouts. In: 5th international symposium on smart graphics; vol. 3638/2005. Berlin, Germany: Springer; 2005. p. 115–26.
[11] Hensley J, Scheuermann T, Coombe G, Singh M, Lastra A. Fast summed-area table generation and its applications. Computer Graphics Forum 2005;24: 547–555.
[12] Java Binding for the OpenGL ⟨http://kenai.com/projects/jogl/pages/Home⟩.
[13] Maass S, Döllner J. Seamless integration of labels into interactive virtual 3D environments using parameterized hulls. In: Proceedings of the 4th international symposium on computational aesthetics in graphics, visualization, and imaging. Lisbon, Portugal: Eurographics Association; 2008. p. 33–40.
[14] Rong G, Tan T-S. Jump flooding in GPU with applications to Voronoi diagram and distance transform. In: I3D '06: proceedings of the 2006 symposium on interactive 3D graphics and games. New York, NY, USA: ACM; 2006. p. 109–16.
[15] Ropinski T, Prassni J-S, Roters J, Hinrichs KH. Internal labels as shape cues for medical illustration. In: Proceedings of the 12th international fall workshop on vision, modeling, and visualization; 2007. p. 203–12.
[16] Scheuermann T, Hensley J. Efficient histogram generation using scattering on GPUs. In: I3D '07: proceedings of the 2007 symposium on interactive 3D graphics and games. New York, NY, USA: ACM; 2007. p. 33–7.
[17] Stein T, Décoret X. Dynamic label placement for improved interactive exploration. In: NPAR '08: proceedings of the 6th international symposium on non-photorealistic animation and rendering. New York, NY, USA: ACM; 2008. p. 15–21.
[18] Vollick I, Vogel D, Agrawala M, Hertzmann A. Specifying label layout style by example. In: UIST '07: proceedings of the 20th annual ACM symposium on user interface software and technology. New York, NY, USA: ACM; 2007. p. 221–30.
[19] Wolff A, Strijk T. The map-labeling bibliography ⟨http://i11www.iti.uni-karlsruhe.de/~awolff/map-labeling/bibliography/⟩.
[20] Yager RR. Fuzzy decision making including unequal objectives. Fuzzy Sets and Systems 1978;1(2):87–95.
[21] Zimmermann H-J. Description and optimization of fuzzy systems. International Journal of General Systems 1975;2(1):209–15.