Temporal Coherence Strategies for Augmented Reality Labeling

Jacob Boesen Madsen, Markus Tatzgern, Claus B. Madsen, Dieter Schmalstieg, Denis Kalkofen
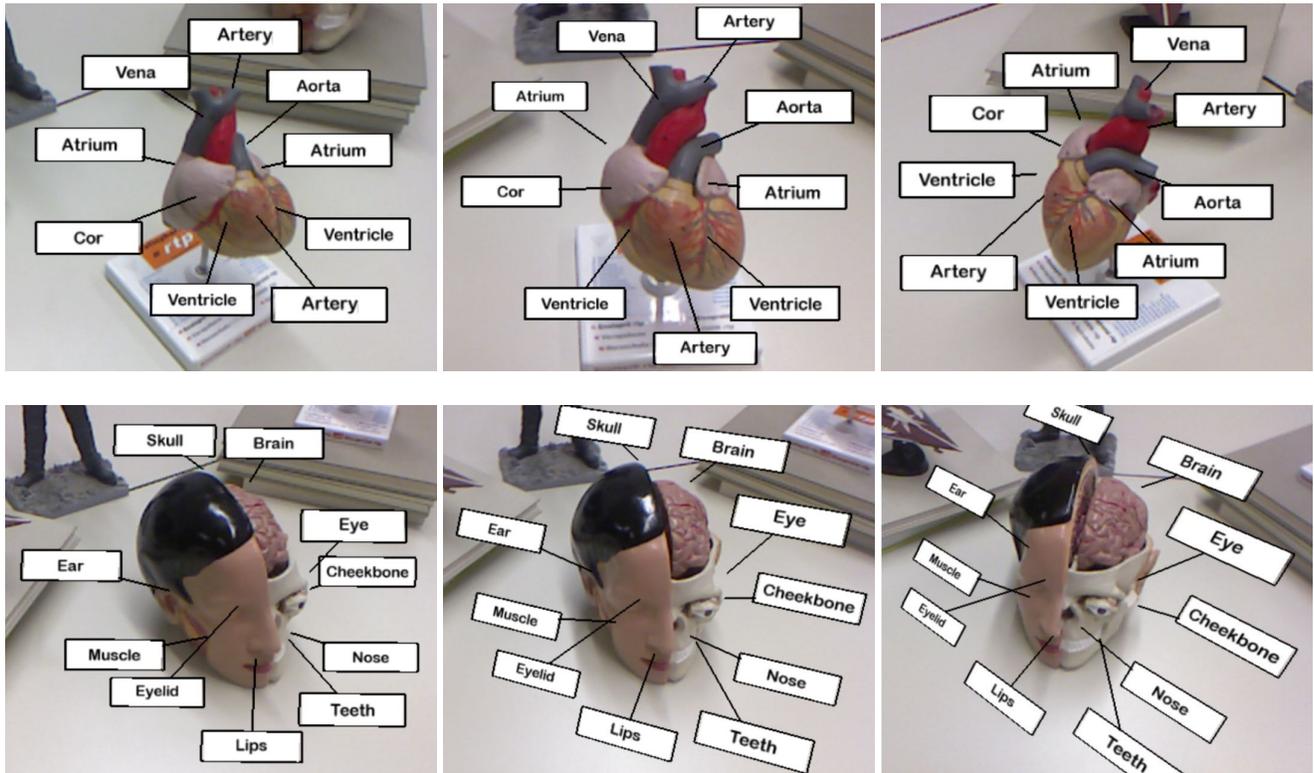
Fig. 1. (top) View management in 2D places annotations in image space and updates it in every frame. Conflicts between labels are resolved, but at the cost of an unstable layout. (bottom) View management in 3D places annotation in a plane defined in object space. This gives the option of making the layout stable by disabling updates, until the change of viewpoint becomes too significant.

Abstract— Temporal coherence of annotations is an important factor in augmented reality user interfaces and for information visualization. In this paper, we empirically evaluate four different techniques for annotation. Based on these findings, we follow up with subjective evaluations in a second experiment. Results show that presenting annotations in object space or image space leads to a significant difference in task performance. Furthermore, there is a significant interaction between rendering space and update frequency of annotations. Participants improve significantly in locating annotations, when annotations are presented in object space, and view management update rate is limited. In a follow-up experiment, participants appear to be more satisfied with limited update rate in comparison to a continuous update rate of the view management system.

---

1 INTRODUCTION

Annotations are commonly used in hand-drawn illustrations to add information, textual or pictorial descriptions to objects. Annotating elements of a map is a thoroughly researched area that has matured towards a common set of best practices in cartography [12]. By harnessing the computational power of modern computers, the placement

---

- *Jacob Boesen Madsen, Claus B. Madsen are with the Aalborg University. E-mail: jbm|cbm@create.aau.dk*
- *Markus Tatzgern is with the Salzburg University of Applied Sciences. E-mail: markus.tatzgern@fh-salzburg.ac.at*
- *Dieter Schmalstieg, Denis Kalkofen are with the Graz University of Technology. E-mail: schmalstieg|kalkofen@icg.tugraz.at.*

of annotations can be fully automatized. View management algorithms automatically generate layouts of annotations for different application cases, such as maps [4] and 3D objects [5].

Numerous automatic view management techniques have been developed that mimic the annotation styles of traditional hand-drawn illustrations [1, 5] by enforcing a defined set of constraints. Hartmann et al. [10] provide guidelines after analyzing a wide range of medical and technical illustrations. They distinguish between two types of object annotations: Internal labels are directly overlaid onto the annotated object, while external labels are displaced from the object to avoid occlusions. Moreover, they identify three constraints for placing external labels: (R1) A label should be placed near the object it refers to. (R2) A label must not occlude another label or an annotated object. (R3) Leader lines connecting labels to annotated objects must not cross.

View management algorithms typically incorporate one or more of these constraints to create layouts of annotations. Aside from creating static layouts of annotations for print media, view management algorithms can also automatically accommodate a changing amount of information and even viewpoint changes of the annotated object.

Hence, annotations can be used in Virtual Reality (VR) to interactively explore a virtual object. The layout adapts accordingly to enforce any constraints that are violated due to the changing viewpoints.

In Augmented Reality (AR), annotations provide additional information about real world objects. Due to the constantly changing viewpoint, the layout must be updated in every frame to resolve constraint violations, making it hard for users to keep track and to focus on single annotations. Therefore, *continuous* view management algorithms must also ensure that the layout respects temporal coherence.

A common approach to achieving coherence is to use hysteresis [5] to delay the positional updates of annotations. Alternatively, one can only update the layout when certain conditions are met, e.g., when the viewpoint of the object changes beyond a certain angular threshold [20, 19]. We refer to such approaches as *discrete* view management methods. Discrete methods trade potentially inferior layouts for improved temporal coherence.

In this paper, we perform a formal user evaluation to compare view management algorithms that continuously update the layout to algorithms that only update the layout at discrete points in time. Our focus lies on AR with permanent viewpoint changes. We limit ourselves to view management approaches that use external labels, since it was shown [6] that external labels are less ambiguous in case of tracking errors. We evaluate common force-based view management algorithms that work in *2D image space* [9], but also in *3D object space* [15, 19].

To the best of our knowledge, the behavior of labels over time has never been part of an evaluation of view management algorithms. Literature usually describes a set of constraints and methods to enforce these by continuously updating the layout. An open question is if such updates have a negative impact on the performance of a user during certain tasks, because they constantly change label positions.

Our intuition was that even though discrete view management algorithms cause violations of the layout constraints during viewpoint changes, they outperform the continuous versions in search-and-select tasks that are typical for AR applications using annotations. Based on our findings, we put forward design recommendations for view management systems.

2 RELATED WORK

Systematic annotation of objects have been discussed by cartographers since the 1970s. Imhof [12] generalizes a set of principles for annotating maps. In the 1980s, Ahn [1] presented an annotation algorithm for annotation of area features, line features and point features on cartographic maps.

With automatic generation of layouts and with digital representation of annotations on a computer, much research has shifted towards implementations of view management systems for digital media. Unfortunately, finding an optimal layout for annotations has been shown to be NP-hard even in 2D [13]. Practical view management systems focus on clustering or heuristic approaches [3, 5, 22], posing layouts as a constrained optimization problem for generating an optimal layout.

Hartmann et al. [9, 10] present guidelines for functional and aesthetic layout of external labels. They propose a set of metrics that can act as positive and negative constraints in a view management algorithm for automatic layout. These metrics are related to readability, unambiguity, aesthetics and frame coherence. Their 2D force-based method is probably the most widely used approach until today. Later work by this group introduced several automatic layout algorithms for external labels [2] and evaluated how coherency strategies can be used to annotate 3D animations of objects [8], such as moving the annotation itself or moving the reference line.

Azuma and Furmanski [3] presented view management techniques of 2D labels for AR based on various clustering strategies. They empirically evaluated user responses to the resulting motion. Further research using a stereoscopic HMD was done by Peterson et al. [14], enabling development of view management algorithms to leverage depth information of the scene to further separate annotations and creating new object annotation scenarios not possible in traditional 2D illustration. They describe a user study evaluating label placement techniques specifically developed for stereoscopic HMD usage, concentrating on
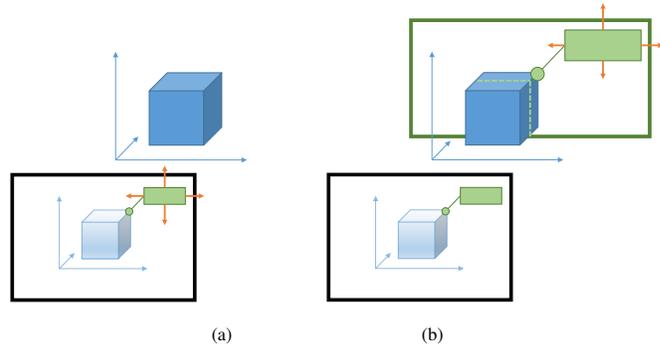

Fig. 2. Consider a scene with a blue 3D cube, which is projected to a light-blue image in the image plane (black frame): (a) View management in image space projects objects first and then places labels in 2D. (b) View management in object space places labels in 3D and then projects them to the image plane.

how depth separation affects response time and errors. Shibata et al. [18] describe the development of a modular view management system for mobile devices. This allows a developer to customize the view management system to target low powered mobile devices for use in mixed reality.

The first 3D view management system for AR was reported by Bell et al. [5]. Their system supports both internal and external labeling with greedy placement. Later work by Pick et al. [15] target an immersive multi-screen environment and resolves 3D occlusions using a technique similar to shadow volumes. Their approach uses a force based approach similar to the one of Hartmann et al., but in object space rather than image space. To ensure legibility of the system, a force is applied to the annotation, ensuring roughly constant distance in object space to the user. They evaluated their implementation using structured expert walkthroughs. Tatzgern et al. [19] developed a system for 3D view management of external labels in object space and addressed the problems of achieving temporal coherency, as the viewpoint changes.

Viewpoint changes trigger label layout updates to resolve violations of the layout rules. To allow users to follow these changes, the positional changes of labels are typically animated. Such animations are commonly used in information visualization to convey state changes to the user during interaction. Hence, animations have been used with the goal reduce the cognitive load when changing the visual states of hierarchical trees [16] or graph visualizations [21]. Heer and Robertson [11] studied animated transitions between different statistical data graphics and found that animated transitions improved graphical perception. The importance of animations is underlined in the design guidelines for fluid interactions in information visualization put forward by Elmqvist et al. [7]. The guidelines explicitly include smooth transitions to visualize transitions between different states so that potentially disorienting abrupt switches are avoided.

View management systems for AR should be able to display annotations coherently, as the viewpoint changes. Essentially, the update of the annotations must not confuse the user. This requirement adds complexity in comparison to static layout methods and has not received much attention in the literature. Ours is the first user study comparing the objective performance and subjective preferences of users exposed to view management in object space or image space. Additionally, we evaluate how discrete or continuous updates affect the users.

3 VIEW MANAGEMENT ALGORITHMS

We begin with a description of design options for view management and the associated advantages and disadvantages.

A two-dimensional label consists of a 2D billboard with text on it, a 3D anchor point on the annotated object, and a leader line connecting the anchor point and the billboard. Annotations can be placed with two degrees of freedom, namely the x and y coordinate in image space,
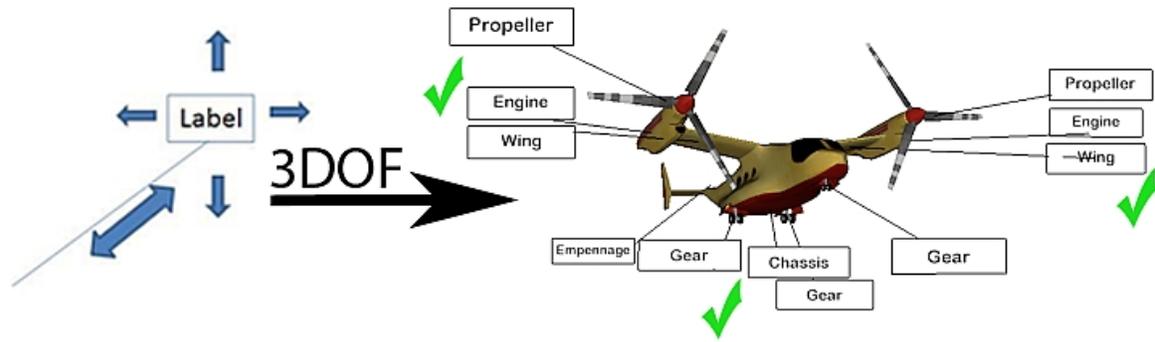
Fig. 3. Placing annotations in 3D relative to the pole attached to the objects yields a useful constraint for temporally coherent viewing. All annotations have been adjusted along the pole first. The annotations marked with a green check-mark have also been slightly displaced in the viewing plane to avoid occlusions.
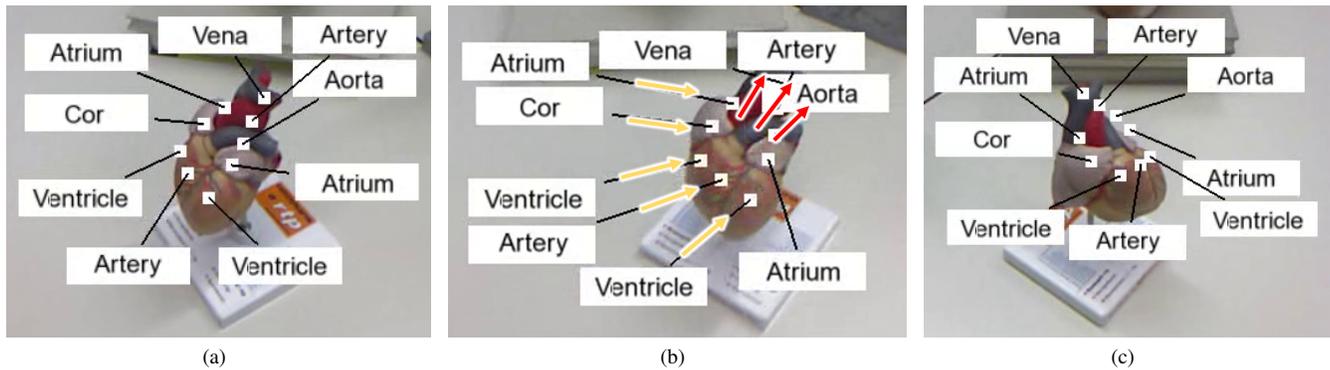

(a) (b) (c)

Fig. 4. Positional drift when using screen-space annotations. (a) Initially, labels are arranged around the object. (b) After camera movement, labels stick to their absolute image location. Note the long leader lines to the left. View management must move the labels back towards the object (yellow arrows) and also resolve the overlaps between the annotated object and the labels (red arrows) (c) Annotations can only stabilize their position relative to the object after the camera stops moving.

since we do not allow label rotation.

A three-dimensional labels consist of similar elements: The billboard is given as a 3D polygon. The 3D anchor point is defined as before. The leader line becomes a 3D pole. Annotation can be placed with three degrees of freedom for position and one or two degrees of freedom for orientation, depending on whether one wants to allow a twist around the viewing vector or not.

3.1 Image-Space vs. Object-Space Layout

Typically, view management techniques describe annotations in 2D relative to the object's projection into the image plane. More precisely, the anchor point is projected to 2D. After this projection, an image-space layout algorithm computes the 2D position of the label, as illustrated in Fig. 2(a). The object (blue cube) is projected to the image plane (black frame), and the label (green) is placed in a 2D location near the object.

Conventional image-space algorithms, such as the one by Hartmann et al. [9] work with forces in 2D. A force resolves collisions (R2) between labels by pushing them away from each other using a direction vector that spans the centers of the 2D labels. The same applies to collisions between 2D labels and the projected 2D bounding box of the annotated 3D geometry. To avoid labels moving too far away from the annotated object (R1), another force pulls the labels back towards its annotated point. Crossing leader lines are resolved (R3) by switching the place of the labels that exhibit crossing leader lines. This is realized by applying a force that is orthogonal to the respective leader line.

Image-space algorithms were designed for producing static images. With camera motion, naive label updates, which re-use the absolute label position from the previous frame, lead to substantial positional drift. Labels stick to the screen, while the object moves away, as can be seen in Fig. 4. Therefore, view management must actively apply rule R1 to move labels closer to the object (yellow arrows) and rule R2 to resolve the overlap between labels and object (red arrows). This positional drift is especially noticeable in AR, where viewpoint changes can include substantial translation.

A better approach for image-space label placement is to store label positions relative to the projection of the anchor point. This requires that the anchor points are re-projected into image space space in every frame, but eliminates the drift problem. Labels will not move further away from the object, but overlaps may still need to be resolved.

View management in object space treats labels as geometry placed relative to the anchor point in 3D. The label is part of the scene, and its projection to image space happens as part of the rendering process. Consequently, no drifting can occur.

Tatzgern et al. [19] propose to place labels in one or more 3D planes intersecting the annotated object, as shown in Fig. 2(b). In this approach, updates to label positions are made in 3D and are guided by 3D rather than 2D constraints.

The pole's orientation is defined by the line connecting the centroid of the annotated object and the anchor point on the annotated object's surface. This ensures that leader lines cannot cross during initial placement, since they emerge radially from the annotated object's centroid.

The billboard must always touch the pole. This allows the following degrees of freedom for the annotation: The billboard can slide arbitrarily along the pole, as long as it does not penetrate the annotated object and does not move further away than a maximum distance from the anchor point. The billboard's center can be rotated arbitrarily about the chosen point on the leader line. Moreover, the billboard can be

displaced arbitrarily in the chosen plane of orientation, as long as it still touches the leader line (Fig. 3). This allows the billboard to be displaced by a maximum corresponding to half of its diameter.

Note that potential violations of the overlap constraint must still be determined in image space. In our approach, we do this by intersecting 2D bounding boxes of objects and labels after projection to the screen.

3.2 Continuous vs. Discrete Updates

Even after resolving the drift issue, perspective projection of labels can still lead to occlusion and crossing leader lines. Resolving these constraint violations requires updates to the layout after camera motion. In a desktop VR setup, this does not cause any problems, since the layout becomes stable, once the user lets go of the camera control. However, in AR, the viewpoint is constantly changing, because the camera is directly attached to the user's hand or head. Locally optimal placement may change from frame to frame even through small unintentional movements of the camera, leading to fluctuations, which are very unnerving for the observer (Fig. 1(top)). Hence, labels never settle, which makes the aspect of temporal coherence a major issue for layout algorithms in AR.

Object-space algorithms can control fluctuations by switching from a continuous to a discrete update strategy. The layout is only calculated for an initial viewpoint. When the user changes the viewpoint, the labels retain their position in object space and remain temporally coherent. If they also retain their orientation (Fig. 1(bottom)), label-to-label overlap cannot occur, but text readability may suffer from perspective foreshortening. If label orientation is re-adjusted to align with the image plane in every frame, label updates have to deal with more occlusions instead.

4 EXPERIMENTAL CONDITIONS

The considerations above suggest a design space for view management algorithms that has two main independent factors: the *space* in which the annotations are described and simulated algorithmically (2D, 3D), and the *update approach* (continuous, discrete). This implies four view management techniques, which we have evaluated in our experiments. We only considered drift-free methods in both 2D and 3D, since drift effects are clearly undesireable and would dominate the experience. For discrete methods, we only compute the layout once and free it after this initialization. For our test scenarios, which have a preferred viewing direction, this is sufficient and avoids handling update rate as a continuous variable.

4.1 Continuous Object-Space Labeling

The continuous 3D algorithm (**C3**) is derived from the force-based method of Tatzgern et al. [19]. Labels are placed in a 3D plane through the object center. Unlike the original method, we update the plane orientation to match the image plane in every frame. Consequently, this algorithm behaves similar to a 2D algorithm [9], and labels are always oriented towards the observer. However, updates of labels postions are made in object space, respecting 3D constraints.
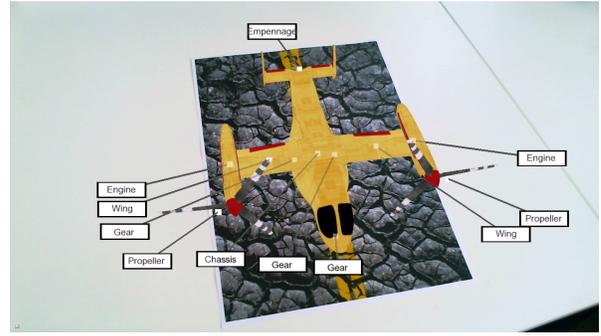
4.2 Discrete Object-Space Labeling

The discrete 3D algorithm (**D3**) works like C3, but calculates the layout only once. After the initialization, the 3D label positions remain fixed. Eventual occlusions between annotations and annotations and the annotated object can be resolved by the user due to the parallax effect of the used 3D planes that place the labels.

### 4.3 Continuous Image-Space Labeling

The continuous 2D algorithm (**C2**) uses a force-based implementation similar to the one described by Hartmann et al. [9], with label positions stored relative to the anchor point projection (2D drift compensation).
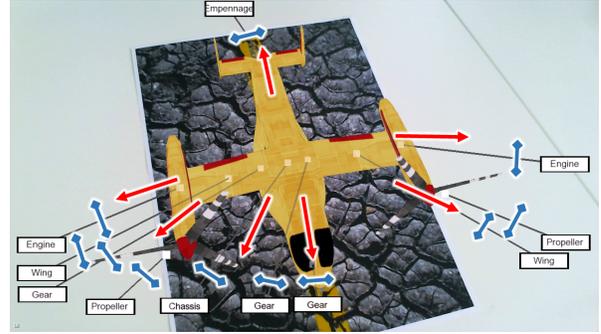
4.4 Discrete Image-Space Labeling

The discrete 2D algorithm (**D2**) is a modified version of C2. If no further updates are made after computing an initial layout, the aforementioned 2D drift compensation can at least compensate for translational camera movements. However, rotations and scale changes quickly

(a)
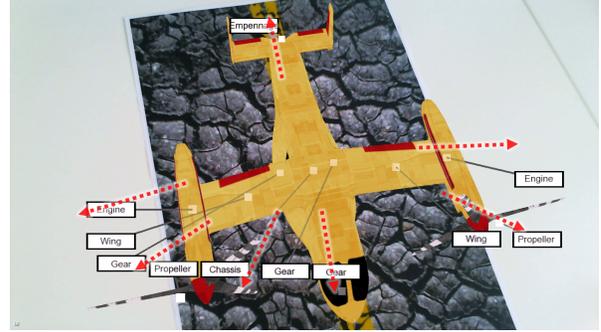
(b)

(c)

Fig. 5. Discrete screen-space labeling. Unlike in the 3D view management, layout updates cannot be completely avoided. (a) The layout for the initial viewpoint. (b) If layout updates are stopped, labels overlap the annotated object when the viewpoint moves closer. The dotted arrows indicate the deactivated constraint for resolving overlaps. (c) The view management system continuously updates the layout to resolve overlaps (arrows). Another constraint ensures that labels stay close to their initial position on the bounding geometry of the object. Small movements are allowed to resolve eventual overlaps between labels (blue lines).

lead to substantial overlap between labels and objects, effectively rendering this method useless. We therefore decided to use a variant of C2, which retains the R1 and R2 forces. We only disable the R3 force resolving crossing leader lines, since, in practice, R3 is responsible for most of the disturbing non-continuous motions. Fig. 5 illustrates these issues using the example of a motion bringing the viewpoint closer to the object.

5 EVALUATION: IMAGE VS. OBJECT SPACE

In this evaluation, we investigate the task performance of the four implementations of the view management system.
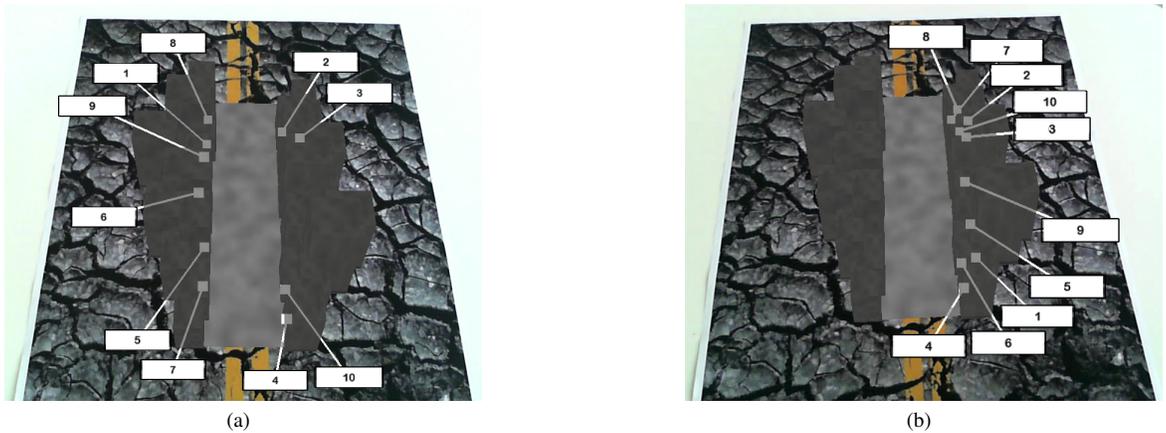
Fig. 6. (a) The distribution of anchor points is balanced over the object. (b) Anchor points are clustered on one side of the model.

5.1 Scenario

The experiment simulates a learning scenario, in which a user is confronted with an unfamiliar object. A user will typically get an overview of the parts from a more distant viewpoint, before stepping closer and investigating the details of the annotated object. Labels give the users an explanation of the annotated parts. They identify a part by following a leader line to the anchor point.

To avoid that participants rely on familiarity with the object of the study, we use an abstract shape that has no resemblance with any real world object. It consists of annotated blocks of approximately equal size and uniform color (Fig. 7(a)). The objects lack any salient clues, which participants could use to memorize the location of the anchor points. Our intention was to make the perceived visual clutter in this experiment is consistent among the participants and avoid influence from prior knowledge of the object or any prior expertise [17] for the objects in this test scenario, given the users unfamiliarity with the scene, task and objects presented.

We use the same 3D positions for anchor points across all experiments to achieve consistent visual clutter. To minimize any bias of individual task and factor combinations, each participant is presented with a randomized selection of anchor points and associated labels, based on the task conditions.

The perceived visual clutter may differ between participants. However, we use an identical setup for each participant, so that the objectively measurable visual clutter in the scene is consistent.

5.2 Apparatus

The experimental code is written in C++ using OpenSceneGraph[1] for task creation and scene display. Marker tracking is handled via a natural feature tracker. The trackable marker is printed on an A3-format paper (297mm×420mm) and is placed as the sole item on a freely standing table, with sufficient room to move around.

The experimental application is deployed on a Microsoft Surface Pro 2 tablet running Windows 8, with an Intel Core i5 CPU, 4GB RAM and a 10.6" screen (1920×1080, 208 ppi, 16:9 aspect ratio). We use the tablet's rear-facing camera (1.2MPixels, 720p) for tracking. Input to the application is handled via the touch screen.

5.3 Study design

We define three independent variables for this study: the update method (continuous, discrete), the spatial description of labels (2D, 3D) and the distribution of anchor points of the object (balanced, unbalanced). The variables regarding the update method and spatial description directly refer to the previously described view management algorithms: 2D image-space with continuous update (C2) and discrete

[1]http://www.openscenegraph.org/

update (D2); 3D object-space with continuous (C3) and discrete update (D3).

We included the distribution of anchor points as variable, because we wanted to investigate its effect on the behavior of labels during on the viewpoint changes. We speculated that multiple anchor points grouped very closely together on the reference object cause more violations of the layout constraints and stronger label movements in continuously updating view management systems. In contrast to such unbalanced layouts (Fig. 6(b)), a more balanced distribution (Fig. 6(a)) of anchor points causes less changes.

The experiment is a mixed methods design, using a randomized, repeated-measures design, with two factors being within-subject, and one factor being between-subject. The within-subject factors are update method (continuous, discrete) and the spatial description (2D, 3D), while distribution of anchor points (B=balanced, U=unbalanced) is a between-subjects factor. The within-subject factors correspond to the four view management systems. Each participant performed three repetitions, leading to a total of 12 tasks per participant. For each participant, the combination of factors and their order was randomized using Latin squares.

As dependent variables, we measured the duration of each task, and the duration of the full trial. Furthermore, we measured error rate metrics and layout statistics: the amount of wrongly identified labels, label order changes, leader line crossings, object space and screen space movement of the relevant labels.

5.4 Task

A task consists of the following steps (Fig. 7):

S1 The participant must identify three labels of interest in a certain order in the overview by clicking on them, then

S2 physically move the viewpoint closer to each anchor point of the corresponding label of step 1.

S3 Repeat (S1) and (S2) three times for each factor-level combination

The purpose of the tasks is to simulate a learning environment, in which a user gets an overview of an object from a viewpoint from which the whole object and its annotations are visible. This is simulated with step (S1), in which the participant had to select a randomly generated sequence of three labels. The sequence was shown on the mobile device (Fig. 7(a)). After identifying and clicking on all the relevant labels, the participants had to physically change the viewpoint of the device and move the viewpoint closer to each identified anchor point (S2). Participants performed the second step for each label in the same order as they were presented in the first step. Before moving closer to a label, they had to click on it again to select it. Clicking on
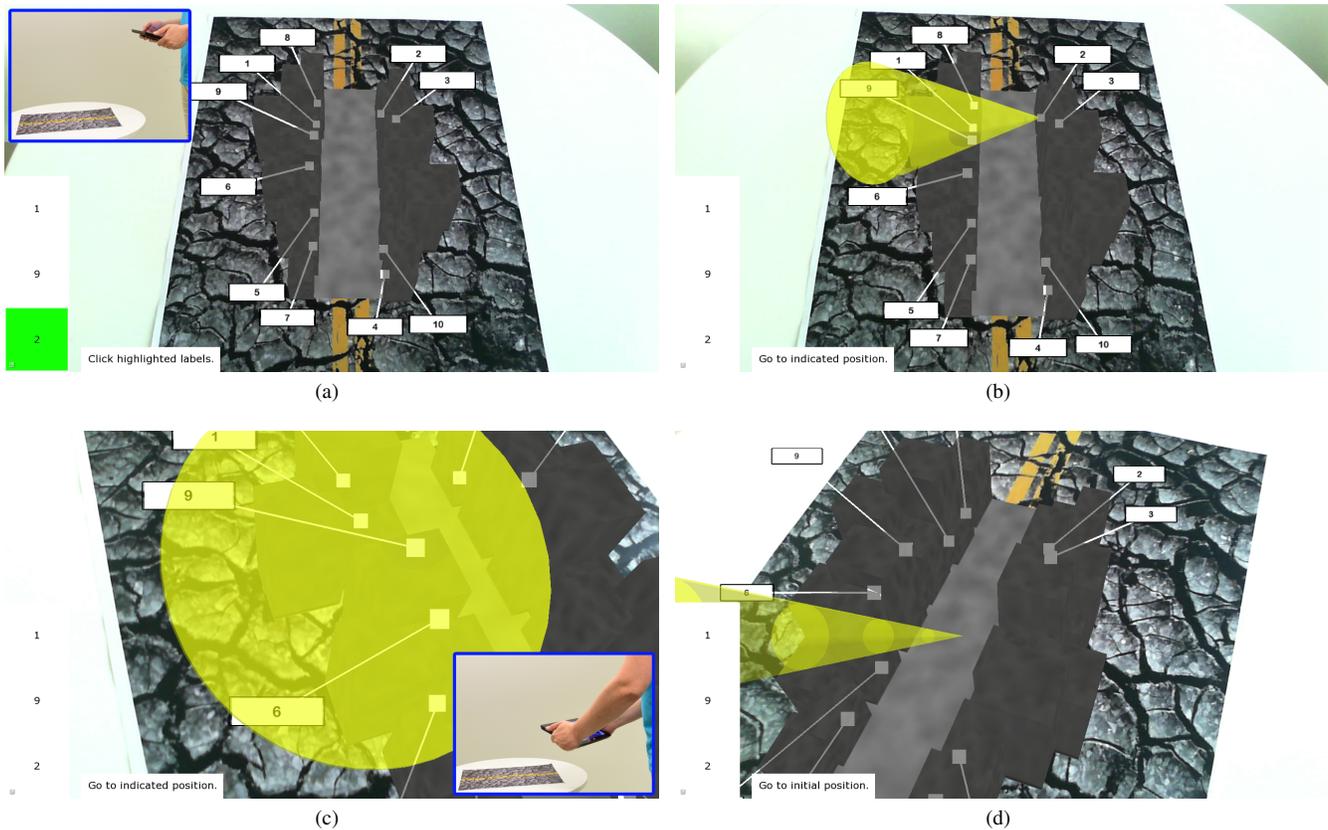
Fig. 7. Experiment participants had to perform the following steps. (a) From an overview position, the participant selects a sequence of labels indicated by the system by highlighting a number on the left. (b) After finishing the sequence, the participant must click on the labels. (c) For each label, the user is asked to align the viewpoint with a cone. (d) After exploring each labels up close, another cone guides the user back to the overview position.

a label would force participants to look for the label by moving the device, which would again trigger layout changes.

After clicking on the label, the system showed a transparent yellow cone, with which the participants had to align the mobile device in order to continue the study (Fig. 7(b)). This additionally enforced movement of the mobile device. The bottom of the cone indicated the position the mobile device should move to, while the tip of the cone pointed to the anchor point of the identified label. The participant had to align the mobile device with the bottom and look at the tip of the cone (Fig. 7(c)). The cone disappeared when the alignment was correct, which indicated that the participant could continue with the task. We introduced a positional and angular tolerance to the alignment, to avoid that participants spend too much time aligning the view. During the trials, we did not experience issues with participants having alignment problems.

After aligning the device with the cone, the task continued with the next label, until the task forced participants to go back to the overview to trigger the next iteration (Fig. 7(d)), starting again with step (S1). Overall, each participant repeated the task twelve times, three times for each investigated view management system.

5.5 Hypotheses

We expected that user performance of task completion differs depending on the view management systems. We had two main hypotheses:

- **H1**: User task completion time differs between the view management systems.

- **H2**: Anchor point distribution has an effect between view management systems.

Regarding H1, we expected that the properties of the view management system influences the task performance during viewpoint changes. When a user changes the viewpoint, a continuously updating system constantly resolves layout constraints. Therefore label positions and their relationship to each other and to the annotated object change. We reasoned that such changes have a negative impact on repeatedly locating labels, as required by the task of this evaluation. In discrete setups, the relative label layout does not change, which makes it easier for users to keep track of the locations of labels during viewpoint changes and improves task performance.

In the discrete case (D2 and D3), labels will never change relative order. Like D3, D2 is prone to layout violations from crossing leader lines. Therefore, we expected the discrete update methods D2 and D3 to outperform both continuous update methods C2 and C3. Hence, the properties of the view management algorithm was considered to have an effect on task completion time.

Regarding H2, we expected that the distribution of labels relative to the object influences the view management systems in different ways. For this purpose, we defined balanced and unbalanced distributions of annotated labels. The unbalanced layout grouped anchor points and their corresponding labels closely together. We speculated that during viewpoint changes, this setup would cause more layout violations and more label updates than a balanced setup, where anchor points and labels are well distributed. We hypothesized that balanced and unbalanced layouts would lead to a performance difference, because the relative locations of labels changed to a different degree.

5.6 Procedure

Prior to starting the experiment, we asked the participant to fill out an informed consent form along with a demographic questionnaire, in-
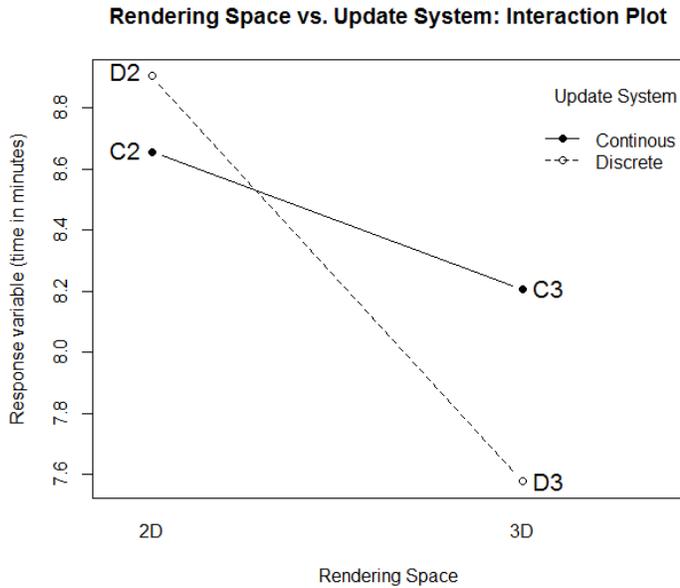
Fig. 8. Interaction plot of factors rendering space and update method. Although significance can be reported, the data suggests an interaction is present between the two factors.

Fig. 9. Isolated data of step S1 of the task for each condition. A bar represents the average time for the "selection" part (S1) of the full task (as is shown in Fig. 7 (a)), reported with standard error.

cluding questions of age, familiarity with technology, mobile devices and AR technology. We introduced the participant to the experiment with a thorough explanation of the purpose of the study and the used system.

Before starting the experiment, the participant performed a set of training tasks with the view management system of the current condition. During this task, the participants were free to ask any questions regarding usage and control of the system. The training task was a simplified version of the real task with only four labels, two of which were part of the selection and identification procedure. The configuration of labels of the training task was different than the configuration of the actual task to avoid unintended learning effects. Following the training task, the participant completed the task without interruption. After finishing the task, participants were allowed to take a break, before moving on to the next view management condition, which again started with a training task.

Participants were told to pay attention to solving the task to the best of their abilities, and not mind the amount of time spent on each task. We logged both completion time and error rate.

After completing all trials, the participant responded to a small open-ended questionnaire and gave additional verbal feedback in an interview with emphasis on the interface and their strategies for completing the tasks.

5.7 Participants

A total of 24 participants (6 female) were part of the experiment, aged 24-36 (M=29,3). All were recruited from on and off the campus area. All participants self-reported normal or corrected-normal vision. Participant familiarity with AR was self-reported to be average on a 5 point Likert rating, ranging from novice to expert user, and familiarity with handheld mobile devices above average using the same 5-point Likert rating scale. Data collected from the experiment comprised of data from 24 participants, each with 12 tasks, resulting in a total of 288 tasks. The average completion time of the experiment per user was 24 minutes (SD = 4 min).

5.8 Results

The analysis has been carried out using the statistical software R, using a significance level of $\alpha = 0.05$. The main analysis method were type III ANOVA and Friedman test, testing the overall difference between the three independent variables. Pairwise comparisons in post-
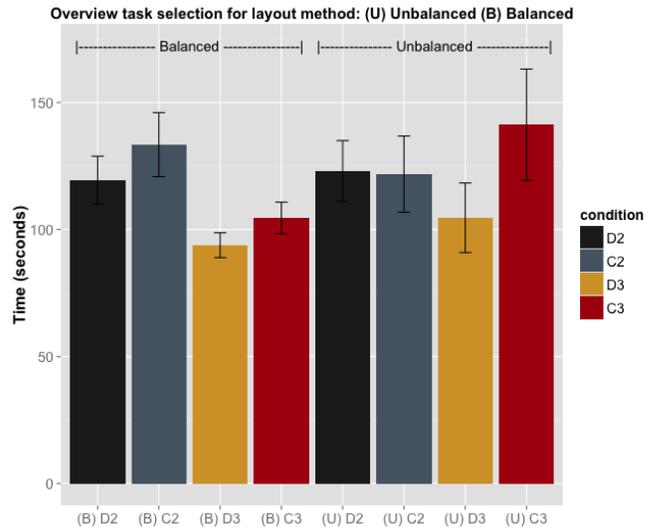
hoc tests were performed using Tukey's honest significant difference (HSD) method for any interaction, while controlling the error rate. Near-significant results reported in the results are defined as being in the range of $0.05 < p < 0.10$.

This section describes different statistical tests throughout. The reason for using multiple different tests lies in the relationship amongst the data. Initially, we log transform the data in order to meet assumptions for normality. However, when splitting up the data or investigating interesting factors or interaction, this proved to be impossible. In these cases, as well as in cases for a single factor, a relevant statistical method has been chosen, according to the data level of measurement, as well as the paramteric or non-parametric nature of the data.

In the experiment, the collected task performance time did not meet the standard assumptions of ANOVA analysis, as the normality of the residuals and the homogeneity of the variance for the factors were found to be problematic. A logarithmic transformation of the task time resolved the problem. Therefore, we used a logarithmic scale for statistical analysis of the task performance time. As verification of the conclusions, a non-parametric Friedman test was run in parallel with ANOVA to ensure that no conflicting conclusions were found and that the conclusions are well supported.

Results of type III ANOVA report the spatial description (2D, 3D) as a significant main effect ($F_{1,22} = 15.79, p < 0.001, \eta_G^2 = 0.084$). This means that participants showed significantly slower completion time in the image-space rendering condition. This is in line with the first hypothesis, H1. The second hypothesis, H2, is not supported in the $log(time)$ data of the full set of tasks, as the layout of anchor points does not show time performance difference between the layouts. No other main effects or interactions were found in the analysis of $log(time)$. However, the two-way interaction between spatial description and update method was near-significant ($F_{1,22} = 3.90, p = 0.061, \eta_G^2 = 0.018$).

The near-significant interaction between rendering space and update method is illustrated in Fig. 8. While the interaction was not significant, there was an interesting visual cue illustrated by the crossing (i. e., interaction), which lead us to investigate possible interactions using a follow-up Tukey HSD test. The test on the within-subjects factors from the experiment showed that the 3D discrete system (D3) significantly differs from both 2D systems, C2 and D2, (both $p < 0.001$). D3 is faster than both C2 and D2. Furthermore, C3 is near-significantly different from D2 ($p = 0.085$). Based on these re-

sults, D3 achieved best task performance in this experiment.

In order to explain the performance difference, we isolated portions of the task for further investigation. In step S1 of the task, participants had to find and select three labels, and repeated this step three times. Therefore, participants could potentially build spatial memory of the label locations, which would be evident in a better performance. To investigate a potential effect on spatial memory, we isolated the data of step S1 (Fig. 9). To investigate the data, we hypothesized that all conditions performed equally, and used a Chi-Squared Goodness-of-Fit to test for consistency in the data, treated as a categorical variable. We found a significant difference from the expected values ($\chi^2(7) = 359.43, p < 0.01$). To determine which factors follow the expected performance, we used multiple pairwise comparisons with Bonferroni correction, which confirmed that only "B+D2", "U+D2" and "U+C2" follow the expected performance, while the rest either has a longer (B+C2, U+C3) or shorter average duration (B+D3, B+C3, U+D3) as Fig. 9 indicates.

Visually inspecting the results, the difference between balanced and unbalanced C3 (red bars in Fig. 9) is larger than for the other systems. We investigated the data of label order changes for C3 non-parametrically for difference in mean using a Mann-Whitney U test, because the isolated data does not meet the criteria for normality. The test indicated that the number of reorderings of labels between the two conditions is a significant factor: The mean ranks of balanced and unbalanced groups were 25.5 and 47.5, respectively ($U = 251, n = 72, Z = -4.4713, p < 0.01$). This result indicates that anchor point distribution might have an effect on view management system in accordance with the H2 hypothesis.

6 EVALUATION: CONTINUOUS VS. DISCRETE

We performed a follow-up study to collect more qualitative feedback on selected view management systems. Although participants of the previous study already filled out a questionnaire to collect qualitative feedback, the questionnaires did not yield any reliable results regarding the preferences of systems. Based on the feedback gathered from participants, we believe that participants could not distinguish between the four systems after completing the experiment.

To avoid users mixing up the different systems, we focused on only two view management systems. The first study identified D3 as the one achieving best task performance. Therefore, we removed the 2D conditions and compared only C3 and D3 in this study. Furthermore, the data from the first experiment indicated that unbalanced layouts cause stronger layout changes than balanced layouts. Therefore, we removed the independent variable regarding the distribution of anchor points of the object by focusing only on the more challenging unbalanced scenario. The apparatus, task and procedure were identical to the first experiment.

6.1 Study design

The study had a randomized, within-subjects design with one independent variable: update method (continuous, discrete). In this study, we only used 3D view management methods. Therefore, the method directly refers to the continuous 3D method (C3) and the discrete 3D method (D3). Participants performed the same task as in the first study, as illustrated in Fig. 7. For the two conditions, this produces a total of six tasks per participant.

Data is procured from a 5-Likert scale questionnaire on the participant's satisfaction using the methods, immediately following the tasks. The scale for satisfaction ranges from dissatisfied to satisfied with the behaviour of the system that had been used. Participant preference data is gathered following the full test of both factors, and the participant is prompted to answer which methods was preferred for the given tasks, forcing the user to make a choice.

6.2 Participants

A total of 10 participants (all male) were part of in the second experiment. All participants where recruited from the same pool of participants as those who took part in the first experiment. No participant took part in both experiments. All self-reported normal or corrected-normal vision.

6.3 Results

The analysis was performed with the statistical software R, using a significance level of $\alpha = 0.05$. The analysis method for user satisfaction was a Wilcoxon signed-rank test. Analyzing participant preference scores was performed using the Exact Binomial test method.

We found a significant effect when analyzing the difference in the responses of the 5-Likert scale question of participant preference. The mean ranks of D3 and C3 were 14 and 7, respectively: $W = 3.5, Z = -2.21, p = 0.02734, r = -0.49$. For ties in the data, where two or more values are the same, we average the ranks for the tied values to compute the values. This is a strong indication that users were overall more satisfied with the discrete update system, in comparison with the continuous, despite a lack of significant differences in task performance.

For participant preference, users were asked to make a binary choice of preference, choosing between either the continuous or discrete update system. As one user did not have a preference, the outcome was eight for discrete, one undecided, and one for continuous. This indicates preference towards the discrete system, even though the small sample size does not allow to observe a statistically significance (8/10 votes for discrete updating is not significantly higher than chance at 0.5, $p = 0.055$, 1-sided. We are confident that the binary choice of preference shows borderline significance due to the small number of participants in this second experiment. A follow-up study with a larger number of participants will provide valuable insights in user preference for continuously or discretely updating systems.

7 DISCUSSION

The first study clearly showed that the view management system, which treats labels as 3D objects and creates a static layout (D3), significantly outperforms the 2D continuous view management system (C2). This is in line with our expectation, because the continuously updating layout seems to make it difficult for users to keep track of the labels. However, D3 also outperforms its 2D counterpart D2, which avoids strong label motions by preserving the order of labels. This difference can be attributed to the more stable placement in object space.

By inspecting the performance data (Fig. 8), we can see that D2 and C2 exhibit very similar performance. This indicates that, even though D2 enforces a certain label order, the small motions of the simulation running in the background and the lack of a static 3D representation may have a negative impact on the ability of users to locate and interact with labels.

The difference between the 2D systems and D3 could also be explained by the way labels are implemented. Despite using force-based approaches for both systems, the spatial representation of labels clearly influences the implementation of the systems. To isolate this factor, we also included a continuously updating 3D layout (C3) in our study. Indeed, the near-significant difference between D2 and C3 hints at implementation specific differences (Fig. 8). This supports our argument that a spatial representation of labels as 3D objects may be preferable. Even though the discrete algorithm D2 does not rearrange labels, it seems to perform worse than an implementation that continously rearranges labels, but works in 3D space (C3). This aspect requires further investigation by unifying the behavior of 2D and 3D layout algorithms based on quantitative analysis of differences in motion patterns. However, due to the difference in the spatial representation, it may be extremely challenging to make the systems behave exactly the same.

Isolating the data of step S1 of the task, (Fig. 9) shows a difference between D3 and C3, which is not present in the initial analysis of the first experiment. We performed a follow-up study, in which we compared D3 and C3 in order to collect more qualitative feedback to investigate this difference. For this study, we focused on an unbalanced label distribution, because this is the worst-case scenario for label placement. Small viewpoint changes can already introduce a reordering of labels. This study revealed that participants preferred

a discrete 3D layout (D3) for the given task. Therefore, we can recommend 3D layouts, which are placed in static locations relative to the object, as the most suitable approach to present labels in a view management system.

Overall, we accept hypothesis H1. A combination of update method (continuous, discrete) and spatial description (3D, 2D) has an influence on task performance. The static layout of the discrete 3D view management system significantly outperforms the 2D versions. A visual inspection of the performance data also shows better performance, when compared to the continuous 3D version. Due to the small sample size of our second study, we refrain from reporting on the performance difference between C3 and D3. To gather reliable performance data, a follow-up study should investigate the performance aspect with a larger sample size.

Regarding the second hypothesis H2, we did not find any significant difference in $log(time)$ task performance between balanced and unbalanced layout systems. Nevertheless, we noticed a larger number of label changes in the data when the label layout was unbalanced. To compare this layout data across all systems, we limit the data set to logged data in which the users are in overview mode, i. e., the part of the task where all labels are visible and the participant must identify three labels correctly.

The number of label order changes in the unbalanced condition is 1151 changes, while the number for the balanced condition is 457. This leads to approximately four changes per overview in the unbalanced condition and between two and three changes per overview in the balanced condition (24 participants × 4 systems × 3 tasks = 288 overviews). Hence, the amount of layout change is higher, when algorithms create layouats for unbalanced label distributions, than for balanced label distributions. This objective measurement supports our assumption that layouts are prone to changes, when anchor points are not distributed well in the annotated scene. Furthermore, visual inspection of Fig. 9 shows a strong performance difference of the continuously updating system C3 between the unbalanced and balanced condition. Nevertheless, the results of the first study indicate that avoiding constant label motion and placing labels relative to the annotated object in 3D, as done by system D3, has a larger impact on performance than avoiding label order changes.

8 CONCLUSION

Based on the results of the studies, we can give the following recommendations for designing a view management system. Given that users perform better with labels that are placed statically in the 3D space relative to the annotated object, view management systems should avoid updating labels after their initial placement from the current viewpoint. Furthermore, labels should be treated as 3D objects and part of the scene. Integrating 3D labels into the scene not only allows the AR system to naturally apply the camera pose to the labels, but also simplifies the design of 3D interaction methods. For instance, a method for manipulating a 3D object in AR can be directly applied to manipulating a label. To avoid frequent changes of the layout, which could influence the ability to interact with labels, the anchor points of labels and the labels themselves should be well distributed in the annotated space.

An additional advantage of a static 3D layout is that it can be calculated by optimizing the overall layout for a single frame. After the initial computation, no additional computational resources are required, because the layout is not continuously updated. This factor can be beneficial for the battery life of mobile devices.

For future work, it is interesting to isolate which factor influences the performance of the participants. Object-space labeling may have performed better due to stable label placement or due to the additional spatial cues from registering the labels as 3D objects in the scene. Another avenue of future work is the update method of the object-space labeling system. Currently, the system updates all labels, when the viewpoint moves beyond a certain threshold. The layout of the new viewpoint may be completely different from the one of the previous viewpoint. Such drastic changes may have a negative effect on relocating labels of the object. To achieve better coherence between these layouts, a better solution would take into account the layout of the previous viewpoint when calculating the layout of the new viewpoint.

REFERENCES

[1] J. Ahn and H. Freeman. AUTONAP: An expert system for automatic map name placement. In *Proceedings of the First International Symposium on Spatial Data Handling*, pages 544–571, 1984.

[2] K. Ali, K. Hartmann, and T. Strothotte. Label layout for interactive 3D illustrations. *Journal of the WSCG*, 13, 2005.

[3] R. Azuma and C. Furmanski. Evaluating label placement for augmented reality view management. In *IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 66–75, 2006.

[4] K. Been, E. Daiches, and C. Yap. Dynamic Map Labeling. *IEEE transactions on visualization and computer graphics*, 12(5):773–780, 2006.

[5] B. Bell, S. Feiner, and T. Höllerer. View management for virtual and augmented reality. In *Proc. ACM Symposium on User Interface Software and Technology (UIST 2001)*, pages 101–110, 2001.

[6] E. M. Coelho, B. MacIntyre, and S. J. Julier. OSGAR: A scene graph with uncertain transformations. *Proc. IEEE International Symposium on Mixed and Augmented Reality (ISMAR)*, pages 6–15, 2004.

[7] N. Elmqvist, A. Vande Moere, H.-C. Jetter, D. Cernea, H. Reiterer, and T. J. Jankun-Kelly. Fluid interaction for information visualization. *Information Visualization*, 10(4):327–340, Oct. 2011.

[8] T. Götzelmann, K. Hartmann, and T. Strothotte. Annotation of Animated 3D Objects. In *In SimVis, SCS*, pages 209–222, 2007.

[9] K. Hartmann, K. Ali, and T. Strothotte. Floating labels: Applying dynamic potential fields for label layout. *Smart Graphics*, pages 101–113, 2004.

[10] K. Hartmann, T. Götzelmann, K. Ali, and T. Strothotte. Metrics for Functional and Aesthetic Label Layouts. In *Smart Graphics: 5th International Symposium, SG 2005*, pages 115–126, 2005.

[11] J. Heer and G. Robertson. Animated transitions in statistical data graphics. *IEEE Transactions on Visualization and Computer Graphics*, 13(6):1240–1247, Nov. 2007.

[12] E. Imhof. Positioning Names on Maps. *The American Cartographer*, 2(2):128–144, 1975.

[13] J. Marks and S. Shieber. The Computational Complexity of Cartographic Label Placement. Technical report, Center for Research in Computing Technology, Harvard University, 1991.

[14] S. D. Peterson, M. Axholt, M. Cooper, and S. R. Ellis. Evaluation of Alternative Label Placement Techniques in Dynamic Virtual Environments. In *Symposium on Smart Graphics*, pages 43–55, 2009.

[15] S. Pick, B. Hentschel, I. Tedjo-Palczynski, M. Wolter, and T. Kuhlen. Automated Positioning of Annotations in Immersive Virtual Environments. In *Proceedings of the Joint Virtual Reality Conference*, pages 1–8, 2010.

[16] G. G. Robertson, J. D. Mackinlay, and S. K. Card. Cone trees: Animated 3d visualizations of hierarchical information. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '91, pages 189–194, New York, NY, USA, 1991. ACM.

[17] R. Rosenholtz, Y. Li, J. Mansfield, and Z. Jin. Feature Congestion : A Measure of Display Clutter. In *ACM CHI 2005*, 2005.

[18] F. Shibata, H. Nakamoto, R. Sasaki, A. Kimura, and H. Tamura. A View Management Method for Mobile Mixed Reality Systems. In *Eurographics conference on Virtual Environments (EGVE)*, pages 17–24, 2008.

[19] M. Tatzgern, D. Kalkofen, R. Grasset, and D. Schmalstieg. Hedgehog labeling: View management techniques for external labels in 3D space. In *IEEE Virtual Reality (VR 2014)*, pages 27–32, Mar. 2014.

[20] M. Tatzgern, D. Kalkofen, and D. Schmalstieg. Dynamic compact visualizations for augmented reality. *IEEE Virtual Reality*, pages 3–6, 2013.

[21] K.-P. Yee, D. Fisher, R. Dhamija, and M. Hearst. Animated exploration of dynamic graphs with radial layout. In *Proceedings of the IEEE Symposium on Information Visualization 2001 (INFOVIS'01)*, INFOVIS '01, pages 43–, Washington, DC, USA, 2001. IEEE Computer Society.

[22] F. Zhang and S. Hanqiu. Dynamic Labeling Management in Virtual and Augmented Environments. In *International Conference on Computer Aided Design and Computer Graphics*, pages 397–402, 2005.