Vision, Modeling, and Visualization (2014)

Real-time Canonical-angle Views in 3D Virtual Cities

T.R. Kol, J. Liao and E. Eisemann

Delft University of Technology, the Netherlands

Figure 1: *Standard top-down views make buildings hard to recognize (left). Our interactive canonical view reveals facades better, without causing confusion or overly unrealistic deformations. Our work facilitates navigation and exploration.*

Abstract

Virtual city models are useful for navigation planning or the investigation of unknown regions. However, existing rendering systems often fail to provide optimal views during the exploration, introduce occlusions, or show the buildings from the top only, which limits the amount of useful visual information accessible to the user. In consequence, users are forced to interact more extensively with the application to avoid these shortcomings. This process can be quite time-consuming. In this paper, we propose a new technique based on canonical views to address these problems. We compute every building's canonical view and, dynamically, transform it correspondingly, so that it is easy to identify under all camera angles. A user study was conducted to assess how this technique compares to a regular view; our method improves the recognizability of the buildings and helps the users explore the virtual city more efficiently. The results indicate that using canonical views is beneficial for efficient navigation in virtual cities.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Viewing algorithms

1. Introduction

3D virtual city models are becoming more prevalent in numerous applications. Virtual cities play a role in the entertainment industry and visualization systems, but also in navigation applications, tourist maps and for disaster-management simulations. An increasing number of tools are available to produce such models (e.g., GOOGLE EARTH), and the number of available models increases constantly.

Nonetheless, most navigation planning tools still rely on combinations of satellite imagery, street-level views, and aerial photographs taken at a 45° angle, as each of these alone has certain shortcomings. Satellite photographs give an excellent overview of the city layout, but buildings are difficult to recognize from the top, and landmarks play a significant role in memorizing and describing a route [Den97]. On the contrary, street-level views fail in giving a global context, but ease recognition. When combined, users need to divide their attention between two windows. Ultimately, 45° aerial photographs seem to lead to a good compromise, but having only four viewing angles can lead to visual clutter

and occlusions of the street in dense areas, hindering successful navigation planning. Similar problems persist for a full 3D visualization, which can still exhibit high visual clutter (Figure 2), or, when moving towards a top-down view, can lead to reduced recognition rates. Furthermore, finding appropriate 3D views can be difficult.

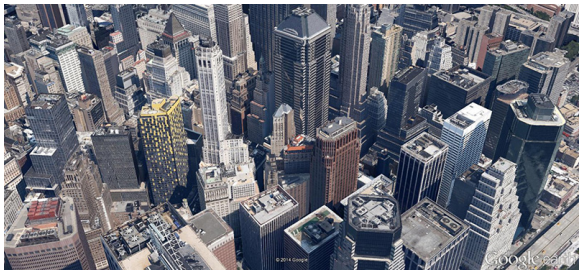


Figure 2: *Manhattan as seen in* GOOGLE EARTH *with 3D buildings enabled. While impressive, note the large amount of visual clutter that prevents users from discerning the street, rendering this view useless for navigation planning.*

For landmark recognition, the viewing angle is important. Tourist maps often rely on iconified versions of the landmarks using a specialized viewing angle, as there is a strong preference for certain viewpoints in the context of object perception [EB92, VB95, BTBV99]. The characteristic viewpoint that users are most comfortable with is called the *canonical view* [PRC81]. Nonetheless, in a dynamic navigation environment, it is not possible to maintain such a view while the user is freely navigating through the scene.

In this paper, we modify the standard perspective to approach a canonical view by ensuring a certain observation angle for each building (Figure 1). Hereby, we facilitate recognition and orientation, making our algorithm useful for navigation tasks. Our work makes the following contributions.

- A viewer exploiting canonical viewing angles for buildings.
- A transformation-conform depth test to avoid sorting.
- A user study to determine preferred viewing angles.

2. Related Work

Obtaining the canonical view of 3D objects [PRC81] has been an active research topic for many years in computer graphics [DCG10, PPB*05] as well as psychology and neuroscience [Pet00]. Originally, Palmer et al. [PRC81] indicated that the quality of a view angle depends on both the visual information that is objectively available, as well as the subjective importance of this information to the user. They found that participants had a strong preference for off-axis views, such as the three-quarter perspective, showing three sides of the object. Congruent results were obtained by [PHL92], who used computer generated images instead of physical objects.

In this work, we focus on building-like structures. Many experiments were conducted with unfamiliar [PHL92], abstract [EB92, CE94] or irrelevant [PH88, HPL91] objects. Even in extensive user studies carried out to determine influential object properties [BTBV99], the model coming closest to the shape of a building was a truck. One conclusion that could be drawn was that the preferred viewing elevation is significantly below 45° and depends on a complex interplay between the geometry, the user's familiarity with the object and the tasks to be performed [BTBV99]. Hence, a universally valid view might not even exist [CE94].

Measures such as *silhouette entropy* and *curvature entropy* [PKS*03], *mesh saliency* [LVJ05], *view entropy* [VFSH01], the *visible area* and *silhouette length* [PPB*05], or *semantics* [MS09] have been proposed to determine best views. Some derivations might be possible for particular objects, but as observable via recognition benchmarks [DCG10, PPB*05], no general solutions seem to exist. Consequently, Secord et al. [SLF*11] focused on a selection of views per object that had been precomputed from a general model, while Yamauchi et al. [YSY*06] proposed to cluster a large set of uniformly sampled viewpoints into several centroids based on their similarity. Our work focuses on buildings, and despite many different appearances, the basic shape is relatively consistent.

In contrast to the previous methods, our goal is not a viewpoint for a compact 3D model, but an optimization of an entire city, in which each building is manipulated to best convey its context and appearance. Other approaches pursued similar goals. Automatically generated tourist maps [GASP08] use optimized views of landmarks to facilitate orientation, but the views and 2D maps remain static. Semmo et al. [STKD12] presented a mix to seamlessly transition between a 3D visualization and a 2D schematic overview of a virtual city. In the 2D view, the landmarks are shown as billboards that transform to regular 3D models when the user zooms in on the map. A different fusion to combine pedestrian and bird's eye views is to bend the city model [LTDJ08]. Furthermore, viewport variations can improve the perception of the spatial relations in 3D [JD08]. While both methods can be useful for navigation, they do not improve landmark views dynamically, which makes investigation tasks more difficult.

A real geometric transformation can make area-of-interest visualizations more successful [MDWK08]. Similar to our approach, they applied a shear transformation on buildings to reveal hidden facades in top-down views next to important streets, while distant geometry remains unchanged. Our transformation however, is dynamically based on the camera and optimizes for a determined viewing angle to produce a more preferable view. In this way, it also differs

from [QWC*09], where landmarks were emphasized along navigation routes, while widening relevant streets to prevent occlusion. While these previous results show the benefit of such modifications for navigation, the production of a single view does not allow free exploration, which is supported in our solution.

3. Canonical Views

Initially, we experimented with different visualization algorithms, but quickly discovered that adhering to the strict definition of a canonical view would restrict interaction and significantly reduce the realism of the resulting rendering, potentially even leading to confusing temporal discontinuities. Therefore, we decided to impose two constraints on the deformation of buildings. First, building footprints should remain fixed to give users a good sense of the structure's location and to prevent floating, which is typical for icon-based maps. Second, we want to enable rotation around the building for exploration purposes, hence we deliberately avoid fixing the view orientation to a three-quarter view. Furthermore, we assume that the effect of the distance to the camera (i.e., the zoom) on the canonical view is negligible, which is similar to the assumptions made for tourist maps.

In consequence, optimal (i.e., preferred) views are derived from a *canonical angle* θ_c per building, which is defined as follows. First, we assume a simpler shape by focusing solely on the bounding box, which is also the shape we will use to derive a preferred viewing angle, making our approach less dependent on attributes such as building styles. The angle of a box is then measured using spherical coordinates with the origin at the center point $\mathbf{c}$ on the top (Figure 3). The goal of our algorithm is to measure this subtended angle based on the bounding box and compare it to the canonical angle, to obtain a corrective elevation offset, which is then transferred to the vertices of the actual building.

Specifically, the vector $\vec{v}$ pointing from $\mathbf{c}$ to the actual camera position $\mathbf{C}$ and the top plane form an angle θ, which optimally should match θ_c. The difference between θ and θ_c is the elevation offset θ_Δ, which will be used to adapt the building to achieve improved viewing conditions. We will assume that the heights of the buildings are along the y-axis and the buildings' floors are parallel to the x,z-plane.

3.1. Building Transformation

To transform the building according to the canonical angle, we first compute the subtended angle θ of the bounding box. Note that this value is negative if the camera is positioned below $\mathbf{c}$, i.e., $\mathbf{C}$ has a smaller y-coordinate than $\mathbf{c}$. Next, by subtracting the canonical angle, we obtain the corrective elevation offset $\theta_\Delta = \theta - \theta_c$.

If θ_Δ is negative, which would correspond to a rotation *towards* the camera, we do not apply our algorithm, as this operation would hide nearby facades, instead of improving the view on the model. If not, we rotate the box by θ_Δ to establish the canonical angle. In other words, given a vertex $\mathbf{p}$ at position $\mathbf{p}_0$, $\mathbf{p}$ is projected on the bounding box *bottom* resulting in $\mathbf{p}'$ (Figure 5b). Then, $\mathbf{p}_0$ is rotated around $\mathbf{p}'$ in the opposite direction of $\vec{v}$, i.e., away from the camera, with a rotation angle of θ_Δ (Figure 5c), leading to $\mathbf{p}_1$. Performing this operation, the building is rotated to match the intended angle, but its base remains at the same location. This transformation respects the first constraint that we imposed earlier; the building footprint should remain at the same location to avoid the impression of the building floating above the ground. Figure 4 illustrates the final result for a building in a city model.

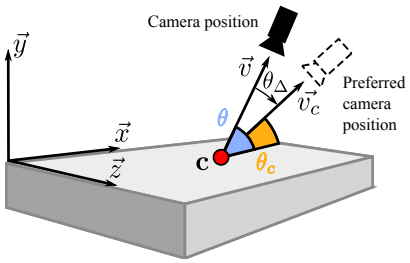


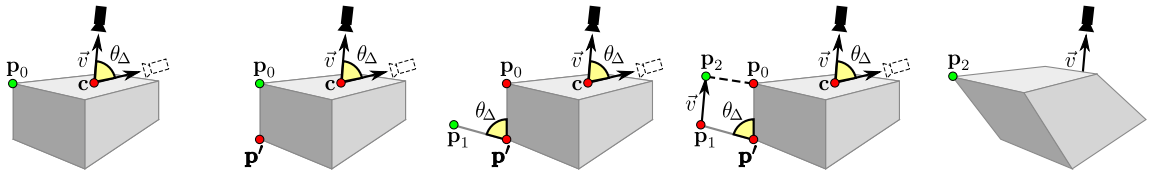
Figure 3: *Pointing from the center of the bounding box top* $\mathbf{c}$ *to the preferred camera position is the vector* $\vec{v}_c$ *with elevation* θ_c*. Vector* $\vec{v}$ *is pointing from* $\mathbf{c}$ *to the actual camera position* $\mathbf{C}$*, having an angle* θ*. We call the difference between these angles the elevation offset* θ_Δ*.*

(a) In normal view. **(b)** All vertices are rotated. **(c)** All vertices are translated.

Figure 4: *Transformation of a building in a city model. Note that this is the same top-down view as in (a), yet the building is transformed in such a way that the viewing angle appears much more comfortable.*

Note that we use the vector $\vec{v}$ pointing from $\mathbf{c}$ to the camera position $\mathbf{C}$ throughout our algorithm. We do not calculate a distinct $\vec{v}$ for every vertex, pointing from $\mathbf{p}_0$ to the camera. This choice is useful to maintain consistency within the building. If we rotate every vertex separately, parts of the object will rotate in different directions, as demonstrated in Figure 6, which can lead to a confusing appearance.

We can establish an interesting link between the previous method and a well-known operation; if we translate $\mathbf{p}_1$ in



(a) Compute θ_Δ. **(b)** Project $\mathbf{p}_0$ on bottom face to get $\mathbf{p}'$. **(c)** Rotate $\mathbf{p}_0$ around $\mathbf{p}'$ to get $\mathbf{p}_1$. **(d)** Project $\mathbf{p}_1$ along $\vec{v}$ to get $\mathbf{p}_2$ at the height of $\mathbf{p}_0$. **(e)** All of the vertices transformed.

Figure 5: *Schematic overview of the algorithm simulating a canonical angle, where the vertex* $\boldsymbol{p}$*, denoted by a green dot and originally at position* $\boldsymbol{p}_0$*, is transformed.*

(a) Using $\vec{v} = \mathbf{C} - \mathbf{c}$. **(b)** Using $\vec{v} = \mathbf{C} - \mathbf{p}_0$.

Figure 6: *Calculating* $\vec{v}$ *per vertex causes deformation.*

(a) Unintuitive intersections may occur. **(b)** We solve this using a modified depth function.

Figure 7: *Intersections may occur for low buildings standing very close to skyscrapers.*

the direction of $\vec{v}$ until its y-coordinate matches $\mathbf{p}$'s original value ($\mathbf{p}_2$ in Figure 5d), only a small deviation is introduced with respect to the rotation angle – stemming from the perspective projection, but the operation becomes a simple shearing. The modification is visually negligible, but makes this operation linear and easy to implement on graphics hardware. Furthermore, this insight will be key in resolving the visibility relationships between the different buildings, as simply using the deformed building's geometry can lead to visual artifacts, as analyzed in the next section.

3.2. Occlusion Test

We have seen that the angle can be optimized by applying a shearing transformation to each building. Nonetheless, using a standard z-buffer can lead to occlusions that are introduced by buildings that might now overlap; especially for low viewing angles (Figure 7a), such situations are common. In theory, all buildings could be sorted and rendered back to front, but this would be costly as it needs to be done per frame, and for the structures within a building, like balconies, one would need to wipe the z-buffer after each building is rendered. In order to solve the visibility problem efficiently, a more careful analysis is needed.

First, we will concentrate on the task of ordering the buildings with respect to each other. Instead of using the standard z-buffer, we redefine the depth function as follows. Conceptually, for each pixel, we cast a ray from the camera to the point on the transformed building, and project this ray along the y-axis onto the x,z-plane. We then find the intersections of this ray with the footprint of the building, and use the distance of the closest intersection point to the camera as a depth function. This process is illustrated in Figure 8. Our transformation algorithm ensures that each ray that reaches a transformed building will, when projected to x,z-plane, intersect an edge of the building's footprint. The only exception occurs when the corrective elevation offset $\theta_\Delta = 0$. This situation can be handled easily by introducing a small extension to the ray to ensure at least one intersection.

It should be noted that we compute the actual footprints of the buildings in a preprocessing step and write them to a file, which is read during initialization along with the mesh file itself. The footprints are generally very lightweight, which makes it easy to maintain real-time frame rates. The building footprints can be passed along to the shader either compactly stored in a texture or as uniform variables. If necessary however, the footprints can always be simplified during the preprocessing step.

This modified depth function enables a correct per-pixel ordering of the buildings for non-overlapping footprints, which is usually the case in city models. If footprints do overlap, they should be considered as one building and fused.

However, this approach obviously produces artifacts due to the fact that the depth function only ensures correct ordering between buildings, without considering the order of the fragments inside a given building.

To solve this problem, we rely on a two-step process. In the first pass, we discover for each pixel which building should be rendered. To this extent, we make use of the modified depth function by computing the distance to the closest

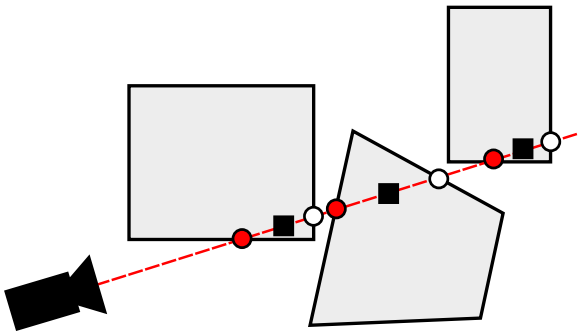


Figure 8: *In this top-down view, the considered fragments for some pixel are shown as black squares. The nearest intersections of this ray with the footprints are shown in red (other intersections are shown in white), and their distance to the camera is used for the depth function. To find these points, a standard algorithm to compute intersections in 2D for line segments is used.*

intersections. The buildings are thus sorted per-pixel according to their relative distance to the camera, and we draw each building with a unique ID to derive a mask that helps us resolve occlusion issues. In the next pass, we make use of the standard z-buffer, but add a check in the fragment shader, in which we rely on the previously determined mask to find out which building the currently drawn pixel should be associated with. If the given fragment comes from a different building, we simply discard it to prevent intersections. Since this second z-buffer test relies on a standard depth function, artifacts are avoided and occlusions are correctly handled, as is shown in Figure 7b.

3.3. Obtaining the Canonical Angle

To find an optimal canonical angle, we relied on a small user study involving eight participants. We showed a bounding box with varying ratios on the screen, resized to fit into a sphere of radius one at the origin. A camera with a 60° opening angle was placed at a distance to the observed object to roughly match the sphere's projection on the screen in pixels. While the box was automatically rotating around the y-axis, we allowed the users to change the angle θ subtended by the bounding box and the camera, as defined previously. Each user performed the test for a total of 64 boxes of varying size ratios, as indicated in Figure 9.

A slight correlation between the canonical angle and dimension ratios can be seen, but, especially for very differing dimensions, we observed a high variance. This result is not surprising, seeing as we rotate the camera around the bounding box and, thus, for these extreme cases, the building can appear very small seen from one side, but extremely stretched from another, producing rather uncertain results. Nonetheless, for larger x/z-ratios, there seems to be a tendency that the preferred camera angle is lowered slightly, which makes sense, as it leads to views that reveal more of the elongated side. Similarly, the canonical angle increases when the object height grows in the y-direction. This observation seems to reflect that the view for tall objects should be more from above to be able to see the whole box.

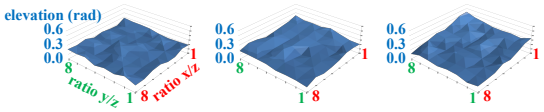


Figure 9: *Results of our user test on the canonical angle. Here, three subjects are shown to illustrate the strong variance. The ratio changes influence the relative height (y) and aspect (x) of the building.*

Overall, we considered the results too noisy to draw general conclusions. In theory, this seems to imply that each user should define their own personalized canonical angle function. Nonetheless, to ease implementation and avoid such a configuration step, we decided to investigate the use of an average canonical angle, namely $\theta_c = 0.273$ radians (15.6°), which already leads to an improved performance in several scenarios (see Section 4.1). Ultimately, a differing preferred angle does not imply that any adaptation is unwanted. It seems that some users simply preferred more drastic degrees of adaptation, which is also illustrated in the user study, showing that the results using the fixed angle are still generally preferred over a standard illustration.

4. Results

The implementation of our algorithm easily reaches real-time rates – only a few basic operations are required per building in the vertex shader at each frame. On a desktop computer with an Intel Core i7 3.7 GHz CPU and a GeForce GTX TITAN GPU, for a city model of 40K triangles with approximately 300 objects, all images were generated in far beyond real-time rates. Results of our algorithm are given in Figure 10 as well as in the various examples showcased throughout the paper. Additionally, in our supplemental video, the smoothness of the transformation can be visually assessed.

4.1. Evaluation

To assess the impact of our algorithm on navigation planning and investigation tasks, we conducted two user studies. The first user study tested the effect of our algorithm on assisting users in finding buildings in a virtual city model, while the second test focused on its effect on memorizing navigation paths within the same virtual city. All tests were performed with the aforementioned equipment.

In total, 24 users with normal or corrected-to-normal vision, age ranging from 23 to 34, participated in our first

Figure 10: *Renderings using the canonical view. We show in each the same scene rendered using normal view (left) and the canonical view (right).*

study, which took around 30 minutes per participant. The second user study featured a comparable group of 12 participants and took around 15 minutes per user.

4.2. Finding Buildings Using the Canonical View

The first task was to find a building in a city model consisting of 300 buildings of different sizes and shapes, as illustrated throughout the paper. The target building to be found was shown in a separate window. Participants were asked to navigate through the city in free camera mode, clicking on the object when they thought they had found it. We timed how long the users took before successfully identifying the object.

In total, users had to perform 12 of these recognition tasks for different buildings at distinct locations. We toggled the canonical view on and off after every task, and varied the canonical view and the order of buildings between participants to ensure unbiased results.

This task turned out to be very demanding. Initially, it is not known in which direction to move, causing participants to move in the direction of, or away from, the target building. This led to longer recognition times for some participants, resulting in a high variance. There was still a 7% improvement in the timings for the canonical view. Nonetheless, the result was not significant due to the high variance.

For this reason, we also conducted a subjective study which was presented to the users right after the experiments. They rated themselves on a scale from 0 to 10 their performance for finding the buildings using the normal and the canonical view. The results of this evaluation are shown in Figure 11, which shows a preference for the canonical view and a relatively low standard deviation. This indicates that users rank their performance significantly higher for our algorithm than for a regular camera model. Furthermore, participants indicated that they were not confused by the viewing algorithm, and that the canonical view felt quite intuitive.

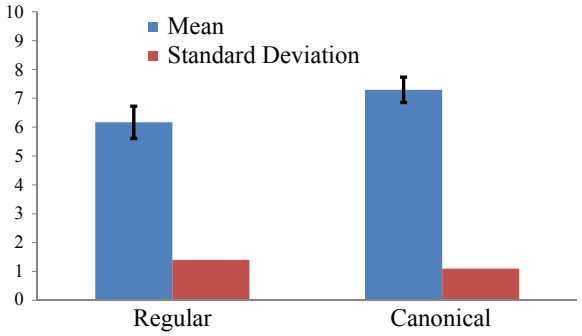


Figure 11: *User rating of all 24 participants on their own performance using the regular and canonical views for finding buildings, showing the means and standard deviations, with confidence intervals of* $p = 0.05$.

4.3. Memorizing Routes

The second evaluation focused on navigation planning. We showed 12 participants a video of a route in our city model two times, after which they were asked to follow the same route themselves, with the camera constrained to the roads. This time, we alternated between a regular top-down view, a canonical top-down view, and a street-level view for the preview videos. In total, three navigation tasks had to be performed, with a different route and preview video every time. We switched the order of routes around for unbiased results.

When users took a wrong turn, we immediately notified them, but also introduced a 5-second penalty. By timing how long it took the participants to arrive at the endpoint of the route, we obtained the results shown in Figure 12. Showing the preview videos in canonical view or street-level view led to slightly better results regarding the navigation time. However, street-level views always have the disadvantage of not showing the whole context. For memorizing a route this may work, but efficiently planning complicated routes is simply not possible, and getting lost might be riskier.

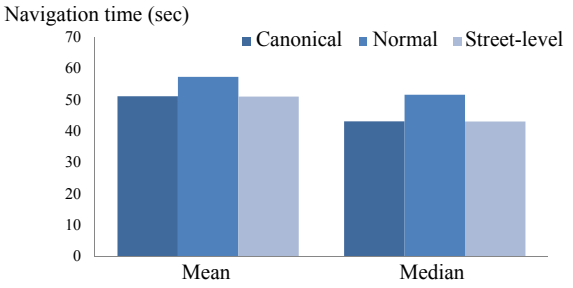


Figure 12: *The mean and median time it took the 12 users to arrive at the endpoint of the memorized route.*

The most interesting result of this user study however, was the amount of wrong turns that were taken given the respective preview videos. We kept track of which participants went the wrong way, and found that for the canonical view, only two users failed to memorize the route. For the street-level view, three out of 12 users got lost, while for the regular top-down view, half of all participants took a wrong turn. This indicates that our algorithm significantly improves the ability to memorize a route in comparison to a standard top-down view. Finally, when asked about their preferred view for these preview videos, half the users opted for the canonical view algorithm, three picked the street-level perspective and one the regular top-down view. Two indicated that they would prefer a combination of the canonical view with either the street-level or the regular view.

4.4. Discussion

The nature of our algorithm causes buildings to shear in a different direction very suddenly when traversing over them from a top-down view. This is due to the fact that the direction of $\vec{v}$ changes immediately. While we experimented with smoothing these transitions, such a solution diminishes the canonical view and results in a limited facade visibility when viewing a building from the top, which was one of the main problems we were aiming to solve. We also investigated shrinking the roofs, but this led to confusing configurations and negative user remarks.

Our current viewing algorithm does not seem to confuse participants and is sometimes even relatively subtle. Only when toggling back to a regular view did some of the users realize that there was an actual deformation applied, but they did immediately notice the loss of visual information. Furthermore, when keeping the camera above the street, all houses will always fold outwards in a top-down view, giving a better overview of the street than with a regular view (Figure 13). In navigation applications, one could imagine constraining the camera to be located above the street to ensure this behavior.

Our algorithm produces transformed objects that are close to the canonical view, but as it is a shearing transformation,

(a) Street in regular view. **(b)** With canonical view.

Figure 13: *Buildings are transformed away from the camera, which always prevents the street from being occluded when the camera is straight above it.*

the roof is only translated. Nonetheless, a rotation of the roof would not be a good solution; when looking at a building in three-quarter view, it would result in deformations at the edges of the structure. Due to our shearing operation, such unwanted deformations are avoided. We compare these approaches in Figure 14.

(a) Additional rotation of the roof. **(b)** Result with our view algorithm.

Figure 14: *Rotating the roof causes deformation at the edges of the building.*

One may think of other, simpler methods to transform the buildings. One option is to simply widen the field of view, however, this causes all objects to appear significantly deformed, and results in a lot of occlusions. Another way is to rotate all vertices around the center of the bounding box's bottom face. We have implemented this method for the sake of comparison; however, it leads to severe deformations, resulting in the buildings being significantly stretched and sometimes even difficult to recognize.

Overall, our approach has several interesting properties. The visualization seems clear and does not cause confusion, the buildings approach canonical views without restricting navigation, a location on a street is never occluded when in focus, and the algorithm is parameterizable, in the sense that users can apply angle preferences (Figure 15).

5. Conclusions and Future Work

We have presented a system to apply canonical views to buildings in 3D virtual cities without restrictions on the viewpoint. The technique dynamically transforms objects in

(a) $\theta_c = 45°$. **(b)** $\theta_c = 15°$.

Figure 15: *The algorithm can be easily parameterized by supplying a canonical angle corresponding to the user's preference.*

real-time based on the current camera position. Our algorithm rotates buildings away from the camera in top-down views, which reveals facades that are otherwise hidden. The transformation is easily parameterizable, allowing users to choose their own preferred canonical angle. User tests indicate that our viewing tool is subtle and does not cause significant confusion, and improves the recognizability of buildings while also being beneficial for navigation planning in the context of route memorization tasks. In the future, we would like to investigate more dimensions, including distance and orientation, when examining the influence of the canonical angle.

Acknowledgements

The authors would like to thank Dr. Jean-Marc Thiery, who provided valuable comments during the undertaking of the research presented here, and all user study participants. This work was partially supported by the FP7 European Project Harvest4D and the Intel VCI at Saarland University.

References

[BTBV99] BLANZ V., TARR M. J., BÜLTHOFF H. H., VETTER T.: What object attributes determine canonical views? *Perception 28*, 5 (1999), 575–600. 2

[CE94] CUTZU F., EDELMAN S.: Canonical views in object representation and recognition. *Vision Research 34*, 22 (1994), 3037–3056. 2

[DCG10] DUTAGACI H., CHEUNG C. P., GODIL A.: A benchmark for best view selection of 3d objects. In *Proceedings of the ACM workshop on 3D object retrieval* (2010), pp. 45–50. 2

[Den97] DENIS M.: The description of routes: A cognitive approach to the production of spatial discourse. *Cahiers de psychologie cognitive 16*, 4 (1997), 409–458. 1

[EB92] EDELMAN S., BÜLTHOFF H. H.: Orientation dependence in the recognition of familiar and novel views of three-dimensional objects. *Vision research 32*, 12 (1992), 2385–2400. 2

[GASP08] GRABLER F., AGRAWALA M., SUMNER R. W., PAULY M.: Automatic generation of tourist maps. *ACM Transactions on Graphics 27*, 3 (2008), 100:1–100:12. 2

[HPL91] HARRIES M. H., PERRETT D. I., LAVENDER A.: Preferential inspection of views of 3-d model heads. *Perception 20*, 5 (1991), 669–680. 2

[JD08] JOBST M., DÖLLNER J.: Better perception of 3d-spatial relations by viewport variations. In *Visual Information Systems*. Springer, 2008, pp. 7–18. 2

[LTDJ08] LORENZ H., TRAPP M., DÖLLNER J., JOBST M.: Interactive multi-perspective views of virtual 3d landscape and city models. In *The European Information Society*. Springer, 2008, pp. 301–321. 2

[LVJ05] LEE C. H., VARSHNEY A., JACOBS D. W.: Mesh saliency. *ACM Transactions on Graphics 24*, 3 (2005), 659–666. 2

[MDWK08] MÖSER S., DEGENER P., WAHL R., KLEIN R.: Context aware terrain visualization for wayfinding and navigation. *Computer Graphics Forum 27*, 7 (2008), 1853–1860. 2

[MS09] MORTARA M., SPAGNUOLO M.: Semantics-driven best view of 3d shapes. *Computers & Graphics 33*, 3 (2009), 280–290. 2

[Pet00] PETERS G.: Theories of three-dimensional object perception-a survey. *Recent research developments in pattern recognition 1* (2000), 179–197. 2

[PH88] PERRETT D. I., HARRIES M. H.: Characteristic views and the visual inspection of simple faceted and smooth objects: Tetrahedra and potatoes. *Perception 17*, 6 (1988), 703–720. 2

[PHL92] PERRETT D. I., HARRIES M. H., LOOKER S.: Use of preferential inspection to define the viewing sphere and characteristic views of an arbitrary machined tool part. *Perception 21*, 4 (1992), 497–515. 2

[PKS*03] PAGE D. L., KOSCHAN A. F., SUKUMAR S. R., ROUI-ABIDI B., ABIDI M. A.: Shape analysis algorithm based on information theory. In *Proceedings of the International Conference on Image Processing* (2003), pp. 229–232. 2

[PPB*05] POLONSKY O., PATANÉ G., BIASOTTI S., GOTSMAN C., SPAGNUOLO M.: What's in an image? *The Visual Computer 21*, 8-10 (2005), 840–847. 2

[PRC81] PALMER S., ROSCH E., CHASE P.: Canonical perspective and the perception of objects. In *Attention and Performance IX*, Long J., Baddeley A., (Eds.). L. Erlbaum Associates, 1981, pp. 135–151. 2

[QWC*09] QU H., WANG H., CUI W., WU Y., CHAN M.-Y.: Focus+context route zooming and information overlay in 3d urban environments. *IEEE Transactions on Visualization and Computer Graphics 15*, 6 (2009), 1547–1554. 3

[SLF*11] SECORD A., LU J., FINKELSTEIN A., SINGH M., NEALEN A.: Perceptual models of viewpoint preference. *ACM Transactions on Graphics 30*, 5 (2011), 109:1–109:12. 2

[STKD12] SEMMO A., TRAPP M., KYPRIANIDIS J. E., DÖLLNER J.: Interactive visualization of generalized virtual 3d city models using level-of-abstraction transitions. *Computer Graphics Forum 31*, 3 (2012), 885–894. 2

[VB95] VERFAILLIE K., BOUTSEN L.: A corpus of 714 full-color images of depth-rotated objects. *Perception & Psychophysics 57*, 7 (1995), 925–961. 2

[VFSH01] VÁZQUEZ P.-P., FEIXAS M., SBERT M., HEIDRICH W.: Viewpoint selection using viewpoint entropy. In *Proceedings of the Vision Modeling and Visualization Conference* (2001), pp. 273–280. 2

[YSY*06] YAMAUCHI H., SALEEM W., YOSHIZAWA S., KARNI Z., BELYAEV A., SEIDEL H.-P.: Towards stable and salient multi-view representation of 3d shapes. In *IEEE International Conference on Shape Modeling and Applications* (2006), pp. 40:1–40:6. 2