Visual Comput (2007)
DOI 10.1007/s00371-007-0182-z

ORIGINAL ARTICLE

Dmitry Sokolov
Dimitri Plemenos

Virtual world explorations by using topological and semantic knowledge

Dmitry Sokolov (✉) · D. Plemenos
XLIM Laboratory UMR CNRS 6172,
University of Limoges, 83, rue d'Isle,
87000 Limoges, France
s@skisa.org, dimitrios.plemenos@unilim.fr

Abstract This paper is dedicated to virtual world exploration techniques. Automatic camera control is important in many fields as computational geometry, visual servoing, robot motion, graph drawing, etc. The paper introduces a high-level camera controlling approach in virtual environments. The proposed method is related to real-time 3D scene exploration and is made of two steps. In the first step, a set of good viewpoints is chosen to give the user a maximum knowledge of the scene. The second step uses the viewpoints to compute a camera path between them. Finally, we define a notion of semantic distance between objects of the scene to improve the approach.

Keywords Scene understanding · Automatic virtual camera · Virtual camera · Semantic distance

1 Introduction

Exploring virtual worlds has become a more and more important research area in computer graphics the last few years, mainly due to continuous development of Internet services. Efficient algorithms for automatic exploration of virtual worlds may help the user in many ways. For example, by allowing him (her) to understand a scene found on the web or to make a guided visit to a virtual museum. In all cases, an automatic exploration could be proposed to the user, taking into account exploration criteria, such as view quality, smoothness of the camera path and so on.

Two main classes of automatic virtual world exploration techniques may be distinguished:

– Global exploration techniques, where the camera remains outside the world to be explored. These techniques allow a global view of a virtual world but may fail in exploration of some never visible details.
– Local exploration techniques, where the camera moves inside the explored world. These techniques allow one to reach all possibly visible details of the world but they do not give a global view of it.

Recently, some authors have proposed *global* exploration techniques but nothing is written on *local* techniques. In all these papers, the main criterion to determine the camera path is based on geometry of the world to be explored: a point of view is better than another one if it allows one to see more geometrical details than the other. If this criterion is generally pertinent for understanding the geometry of a scene, it is not well adapted to virtual visits of museums or other sites where the objects to see are much more important than the geometry of the site.

In this paper new *local* exploration methods are proposed along with improvements, taking into account semantic information. First, we propose a new technique of automatic camera control, which takes into account scene geometry only. The technique allows a camera to navigate inside a scene until most of interesting reachable places are visited. Then, we describe how the exploration may be improved using additional knowledge on a scene.

The rest of the paper is organized as follows: in Sect. 2 a study of existing techniques in virtual world exploration is presented. Section 3 discusses how to choose a set of good views. Section 4 explains how to create a dynamic exploration based on this data. Section 5 shows how artificial intelligence techniques could be used in the domain of automatic exploration. Finally, in Sect. 6 a conclusion is given with a brief description of future work.

2 Background

With advances in 3D model acquisition technologies, databases of 3D models are evolving to very large collections. Accordingly, the importance of automatically crafting best views, views maximally showing the most important features of an object, has also grown. A number of papers have addressed the problem of automatic viewpoint selection.

2.1 Viewpoint selection

2.1.1 Low-level viewpoint quality estimation

Kamada and Kawai [11] consider a camera position as a good viewpoint if it minimizes the number of degenerated faces when the scene is projected to the screen (refer to Fig. 1). The technique is interesting for small wireframe models, but it is not very useful for more realistic scenes. Since this technique does not take into account which parts of the scene are visible, a big element of the scene may hide all the rest in the final image.

Plemenos and Benayada [14, 16, 17] have proposed a heuristic that extends the definition given by Kamada and Kawai. The method was developed and implemented in 1987 but it was published only in 1991. The criterion is the number of visible details combined with the projected area of the visible parts of the scene. According to Plemenos' definitions, the viewpoint quality can be computed by the following formula:

$$I(p) = \frac{\sum_{i=1}^{n} \left\lceil \frac{P_i(p)}{P_i(p)+1} \right\rceil}{n} + \frac{\sum_{i=1}^{n} P_i(p)}{r}, \qquad (1)$$

where:

1. $I(p)$ is the viewpoint quality for the given viewpoint p,
2. $P_i(p)$ is the number of pixels corresponding to the polygon number i in the image obtained from the viewpoint p,
3. r is the total number of pixels in the image (resolution of the image),
4. n is the total number of polygons in the scene.
5. $\lceil a \rceil$ means the ceiling function, i.e the smallest integer number $a_c \in \mathbb{N} : a_c \geq a$.

The first term in Eq. 1 gives the percentage of visible polygons, while the second term is the percentage of used screen area. Indeed, $\left\lceil \frac{P_i(p)}{P_i(p)+1} \right\rceil$ is 0 if the polygon P_i is not visible from the viewpoint p and 1 otherwise. In other words,

$$I(p) = \frac{\#\, visible\ polygons}{\#\, polygons} + \frac{highlighted\ area}{screen\ resolution}$$

In practice, the viewpoint quality may be computed using GPU. The scene is rendered from a viewpoint, coloring each face with a unique color ID and using flat shading. OpenGL allows one to obtain a histogram which gives an information on the number of displayed colors and the ratio of the image occupied by each color. See [1] and [13] for more detailed description. Figure 2 illustrates the technique. Thus, Eq. 1 becomes:

$$I(p) = \frac{\#\, visible\ colors}{\#\, used\ colors} + \frac{highlighted\ area}{screen\ resolution}$$

Sbert et al. [7, 8, 18, 22, 23] introduced an information theory-based approach to estimate the quality of a viewpoint. Their method evaluates the amount of information captured from a given point of view. The criterion is Shannon's entropy, where projected areas of faces are taken as a discrete random variable. Thus, the maximum entropy is obtained when a certain point can see all the faces with the same relative projected area P_i/P_t.

$$I(p) = \sum_{i=0}^{n} \frac{P_i}{P_t} \cdot \log_2 \frac{P_t}{P_i}, \qquad (2)$$

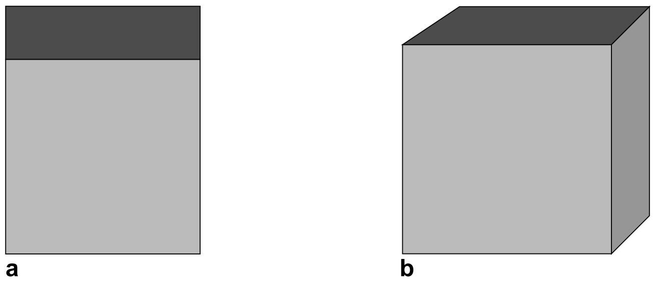

Fig. 1a,b. View **b** is better than **a**, because it does not contain degenerated faces

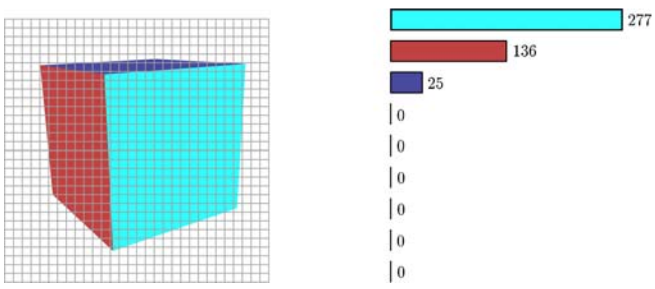

Fig. 2. Detecting visible faces by rendered image analysis. *Left image* represents a rendered cube, *right image* shows the histogram which tells us how many pixels are occupied by each face of the cube

where:

1. p is the viewpoint,
2. n is the number of faces of the scene,
3. P_i is the projected area of the face number i,
4. P_0 is the projected area of background in open scenes,
5. P_t is the total area of the projection.

Later, Sbert et al. [19] have proposed an algorithm, based on the Kullback–Leibler distance. Its objective is to find the minimum representative set of views for a given object or scene.

The above methods give interesting results, but they have some drawbacks. For example, if we consider a very simple scene that consists of one square (Fig. 3a), then the viewpoint entropy equation gives us $I(p) = 0$ for a viewpoint p lying on the perpendicular to the square's center. If we subdivide the square (Fig. 3b), the topology of the scene does not change, but $I(p)$ grows.

In order to avoid this drawback, we have proposed [20] to consider the total curvature of visible surfaces as the criterion of viewpoint quality:

$$I(p) = \sum_{v \in V(p)} \left| 2\pi - \sum_{\alpha_i \in \alpha(v)} \alpha_i \right|,$$

where $V(p)$ is the set of visible vertices of the scene and $\alpha(v)$ is the set of angles adjacent to the vertex v. Figure 4 illustrates the formula. Total curvature of a mesh is the sum of vertex curvatures.

Chang Ha Lee et al. in [12] have introduced the idea of mesh saliency as a measure of regional importance for graphics meshes. Their notion of saliency is inspired by low-level human visual system cues. They define mesh saliency in a scale-dependent manner using a center-surround operator on Gaussian-weighted mean curvatures. The human-perception-inspired importance measure computed by the mesh saliency operator gives more pleasing results in comparison with purely geometric measures of shape, such as curvature.

Blanz et al. in [3] have conducted user studies to determine the factors that influence the preferred views for 3D objects. They conclude that selection of a preferred view is a result of complex interactions between task, object geometry, and object familiarity. Their studies support visibility (and occlusion) of salient features of an object as one of the factors influencing the selection of a preferred view. Gooch et al. in [10] have built a system that uses art-inspired principles and some of the factors suggested by Blanz et al to automatically compute initial viewpoints for 3D objects.

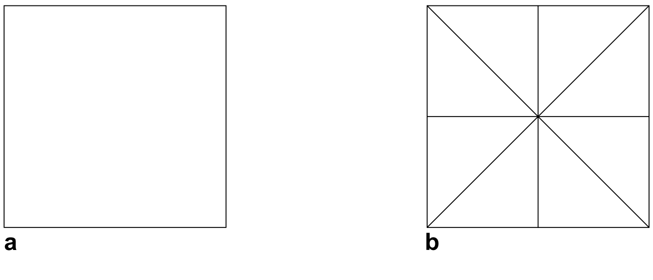

Fig. 3a,b. Viewpoint entropy equation gives different results for the same scene and viewpoint. $I(p) = 0$ for the single polygone scene (**a**), $I(p) = \log 8$ for the subdivided scene (**b**)

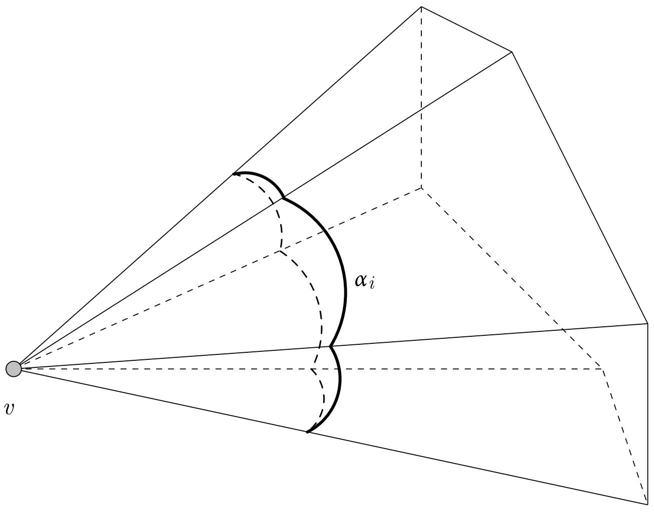

Fig. 4. Curvature in vertex is equal to the sum of angles adjacent to the vertex minus 2π

2.1.2 High-level viewpoint quality estimation

All the methods described in the previous section are low-level methods. While they work well in estimating viewpoint quality of a single object, they often fail in complex scenes. When we say "low-level", we do not want to insult the methods, we mean that the methods use low-level data for input information.

The first high-level method, where a new level of abstraction was added, is proposed by the authors in [21]. The method uses proper (in human perception) division of a scene into a set of objects to improve the notion of viewpoint quality. Figure 5 shows us an example of such a scene, where the subdivision into a set of objects is shown by different colors. These objects are the computer case, the display, the mouse, the mouse pad, two cables, the keyboard and several groups of keys.

Only 20% of the display surface is visible, but it does not embarrass its recognition, because if we are familiar with the object, we can *predict* what does it look like. Thus, we conclude that if there exists a proper subdivision of a scene into a set of objects, the visibility of the objects could bring more information than just the visibility of the faces, and this leads us to the new level of abstraction.

The requirement to divide a scene into objects does not limit the area of the method application. There are many ways to get it. First of all, complex scenes often consist

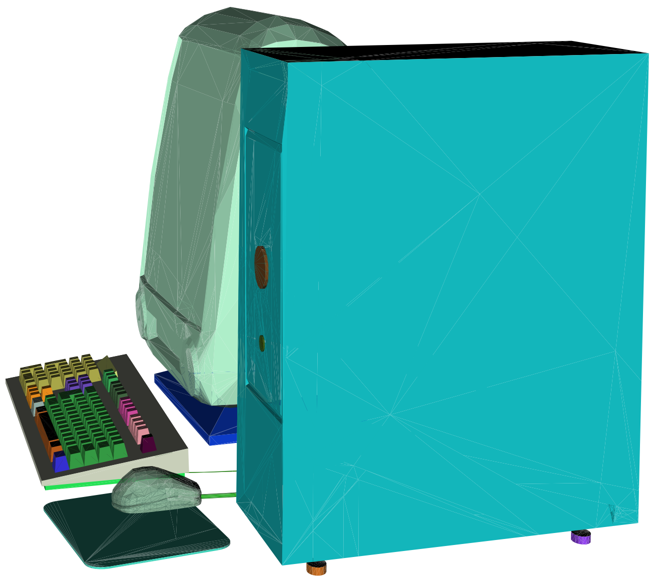

Fig. 5. The scene is subdivided into a set of objects. The display is almost completely hidden by the case, but we clearly recognize it

of non-adjacent simple primitives, and this leads to the first disjunction of a scene. Otherwise, if a scene (or some parts of a scene) is represented by a huge mesh, it could be decomposed. The domain of surface decomposition is well-studied and there are a lot of methods giving excellent results. One can point at results of Zuckerberger et al. [26] and Chazelle et al. [4].

The method could be also very useful in declarative modelling by hierarchical decomposition (refer to [15] for the main principles). In such a case, the decomposition could be provided by a modeler directly, or, probably, it can be combined with the information extracted from a scene geometry.

The main idea is to define how *pertinent* each object is and how well it is *predictable*. Then for any viewpoint we can determine visible parts of the scene. With this information given we can calculate how well each object is observed. The total viewpoint quality consists of the sum of observation qualities for each object.

An accurate definition of the new heuristic is given further. Let us suppose that for each object ω of a scene Ω its importance $q(\omega) : \Omega \to \mathbb{R}^+$ is specified.

We would like to generalize the method and do not want to be limited by a strict definition of the importance function, because it could be done in different ways, especially, if some additional knowledge about a scene is supplied. For example, if the method is incorporated into some dedicated declarative modeler, the importance of an object could be assigned in dependence on its functionality. Moreover, after the first exploration the importance could be rearranged in a different manner to see various parts of a scene more precisely than during the previous exploration.

If no additional information is provided and the user takes into account scene geometry only, then the size of object bounding box could be considered as the importance function:

$$q(\omega) = \max_{u,v\in V_\omega} |u_x - v_x| + \max_{u,v\in V_\omega} |u_y - v_y| + \max_{u,v\in V_\omega} |u_z - v_z|,$$

where V_ω is the set of vertices of the object ω.

It is also necessary to introduce a parameter characterizing the *predictability* (or *familiarity*) of an object: $\rho_\omega : \Omega \to \mathbb{R}^+$. In other words, the parameter determines the quantity of object surface to be observed in order to well understand what the object looks like. If an object is well-predictable, then the user can recognize it even if he sees a small part of it. Bad predictability forces the user to observe attentively all the surface.

The ρ_ω parameter sense could be also interpreted in a different manner. Even if an object is well-predictable (for example, it is a famous painting), the parameter ρ_ω could be chosen as for an object with bad predictability. This choice forces the camera to observe all the painting.

We propose that one consider the function $f(t) = \frac{\rho_\omega+1}{\rho_\omega+t} t$ as the measure of observation quality for an object with predictability ρ_ω, where $0 \le t \le 1$ is the explored fraction of the object (for example, the area of the observed surface divided by the total area of the object surface). Refer to Fig. 6 for an illustration. If the percentage t for the object ω is equal to zero (the user has not seen the object at all), then $f(t)$ is zero (the user cannot recognize the object). If all the surface of the object ω is observed, then $f(t)$ is 1, the observation is complete.

If nothing is known about a scene except its geometrical representation, then one may suppose that the scene does not contain extraordinary (very rough) topology. Then the

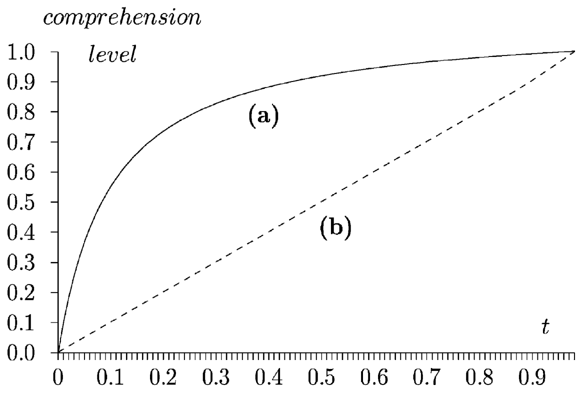

Fig. 6. The behavior of the function $f(t) = \frac{\rho+1}{\rho+t} \cdot t$ for two values of the parameter ρ. (a) $\rho = 0.1$, even a part of an object provides a good knowledge. (b) $\rho = 1000$, the user should see all the object to get a good knowledge

parameter ρ could be taken as rather small value, for example, $\rho_\omega \equiv 0.1\ \forall \omega \in \Omega$. In such a case even exploration of a part of an object gives a good comprehension.

Now let us suppose that there exists some additional knowledge, for example, a virtual museum is considered. Then for all the paintings the parameter could be taken equal to 1000, and for all the walls, chairs, doors equal to 0.1. Now, in order to get a good comprehension of a painting, one should observe all its surface, but only a small part of door or wall is necessary to recognize them.

Let us consider a viewpoint p. For scene objects this point gives a set of values $\Theta(p) = \{0 \le \theta_{p,\omega} \le 1, \omega \in \Omega\}$, where $\theta_{p,\omega}$ is the fraction of visible area for the object ω from the viewpoint p. $\theta_{p,\omega} = 0$ if the object is not visible and $\theta_{p,\omega} = 1$ if one can see all its surface from the viewpoint p.

The fraction $\theta_{p,\omega}$ may be computed in various ways. The simplest one is to divide the area of the *visible* surface by the *total* area of an object. Another way is to divide the curvature of the visible surface by the total curvature of the object. In both cases, we obtain the fraction equal to 0 if an object is not visible at all and equal to 1 if we could see all its surface.

Thus, we propose to evaluate the viewpoint quality as the sum of scene object *importance* with respect to their *visibility* and *predictability*:

$$I(p) = \sum_{\omega \in \Omega} q(\omega) \cdot \frac{\rho_\omega + 1}{\rho_\omega + \theta_{p,\omega}} \theta_{p,\omega}. \tag{3}$$

2.2 Camera trajectory

In practice, a single good viewpoint is generally not enough to understand a complex scene, and even a set of good viewpoints does not allow the user to well understand the scene, as frequent changes of viewpoint may confuse him (her). To solve this problem, virtual world exploration methods were proposed. An important problem in automatic virtual world exploration is to make the camera able to visit the world by using good points of view and, at the same time, by choosing a path that avoids brusque changes of direction.

In [1, 2] an initial idea of D. Plemenos and its implementations are described. The main principle of the proposed virtual world exploration technique is that the camera's movement must apply the following heuristic rules:

- It is important that the camera moves on positions which are good viewpoints.
- The camera must avoid fast returns to the starting point or to already visited points.
- The camera's path must be as smooth as possible in order to allow the user to well understand the explored world. A movement with brusque changes of direction is confusing for the user and must be avoided.

In order to apply these heuristic rules, the next position of the camera is computed in the following way:

- The best point of view is chosen as the starting position for exploration.
- Given the current position and the current direction of the camera (the vector from the previous to the current position), only directions insuring smooth movement are considered in computing the next position of the camera (Fig. 7).
- In order to avoid fast returns of the camera to the starting position, the importance of the distance from the starting to the current position is made inversely proportional to path of the camera from the starting to the current position (Fig. 8).

Thus, the following evaluation function is used to evaluate the next position of the camera on the surface of the sphere: $w_p = \frac{I(p)}{2} \cdot (1 + \frac{d_c}{p_c})$. In this formula:

- w_p is the weight of the current camera position,
- $I(p)$ is the global evaluation of the camera's current position as a point of view,

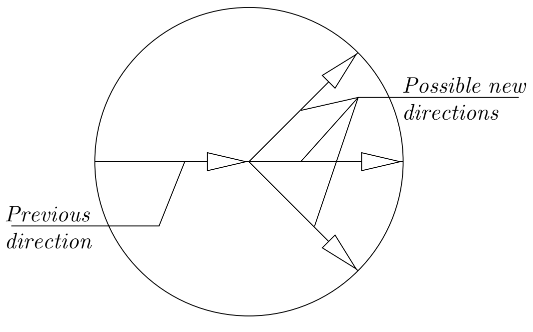

Fig. 7. Only 3 directions are considered for a smooth movement of the camera

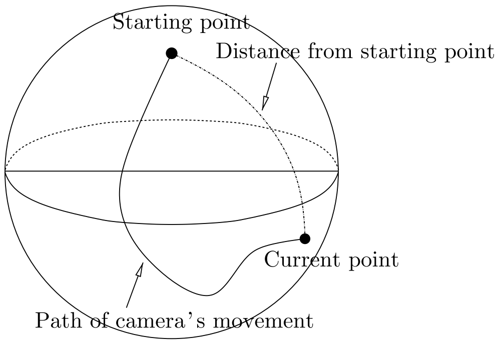

Fig. 8. Distance of the current position of the camera from the starting point

– p_c is the path traced by the camera from the starting point to the current position,
– d_c is the distance of the current position from the starting point.

Vázquez et al. [22, 24] use a similar method for outside and indoor exploration of a virtual world. They use the viewpoint entropy to compute the pertinence of a view.

3 Automatic camera placement

In this paper we propose methods of local scene exploration. The objective is to navigate through the virtual environment to progressively show all the available information to the user. Our goal is to create walk-throughs that simulate human paths, thus, the camera is restricted to be at a constant height from the floor. This section describes a method to choose an ensemble of good viewpoints, which we shall refer as "photos". When the photos are not elucidative, a dynamic method of exploration, proposed in the following section, could be used. It uses the set of "photos" as the control set and creates a walk-through connecting the given viewpoints.

As the viewpoint quality measure we use Eq. 3. The heuristic is very flexible and gives good results even (and especially) for complex scenes. The "photos" should be as good as possible (provide as much information as possible) and the number of photos should not be very great. These criteria are satisfied with a greedy search scheme. The main idea is to define the quality of a set of viewpoints ("photos"). Then, starting from the best viewpoint, at each step we find the best viewpoint for the unexplored part of the scene. The algorithm stops when the quality of the set surpasses the desirable level of the comprehension.

3.1 Observation quality for a set of views

Let us give a more strict formulation. Let us suppose that there is a scene, the set of vertices V and its disjunction into the set of objects are given: $\Omega = \{\omega_k, 1 \le k \le n_\omega\}$, $V = \bigcup_{k=1}^{n_\omega} \omega_i, k \ne l \Rightarrow \omega_l \cap \omega_k = \emptyset$.

As we have said before, the camera is restricted to be at a constant height from the floor. The methods are designed for environments with flat (or almost flat) terrains. In case when the ground is strongly deformed, the scene could be decomposed. In order to reduce the amount of computations, the set of viewpoints is discretized.

Thus, a set of viewpoints S is provided, and a bipartite analytic visibility graph $G = (S \cup V, E)$ is calculated. The set of arcs E represents visibility between objects from S and V. The fact that the arc (s, v) belongs to E means that the vertex v is visible from the viewpoint s. Note that it is sufficient to calculate the point-to-point visibility between vertices and viewpoints, since Eq. 3 uses the intrinsic curvature to compute visible fractions. We have presented [20] a very rapid algorithm to calculate the graph.

Each photograph is determined by three parameters: camera position, view direction and the angle of view. In order to simplify calculations, we have discretized the set of camera positions S. Now we discretize the set of possible view directions. We use a set D of predefined view directions (refer to Fig. 9 for illustration). The set D consists of 65 vectors with the start points in the origin and the endpoints lying in the unit hemisphere.

Therefore, a set of photos is determined by the view angle α and a set of pairs $(s, d) \in S \otimes D$, where s is the camera position and d is the view direction. Let us introduce the notation of a vector: if there is a point p, then $\boldsymbol{p}$ is the vector from the origin to p. Having given a photograph parameters s, d and α, let us find the set $V_\alpha(\{s, d\})$ of visible (in the photo) vertices. The vertex $v \in V$ is visible from the viewpoint s in the view direction d, if the arc (s, v) exists in the graph G and the vector $\boldsymbol{v} - \boldsymbol{s}$ stays in the cone with the axis $\boldsymbol{d}$ and the angle α:

$$V_\alpha(\{s, d\}) = \left\{ v \in V | (s, v) \in E, \cos\frac{\alpha}{2} \le \frac{\boldsymbol{d} \cdot (\boldsymbol{v} - \boldsymbol{s})}{\|\boldsymbol{d}\| \cdot \|\boldsymbol{v} - \boldsymbol{s}\|} \right\}.$$

Now let us extend the definition given by Eq. 3 and define the quality of a set of views (photos). Let us suppose we have selected a set of photos X from the all possible pairs of camera positions and view directions $S \otimes D$. Each object ω of the scene may be visible from multiple photos, so it is necessary to determine its visible part $\theta_{\alpha,X,\omega}$. Let us denote the curvature in a vertex $v \in V$ as $C(v)$ and the total curvature of a mesh $V_1 \subseteq V$ as $C(V_1) = \sum_{v \in V_1} C(v)$. Then for a given object ω its visible fraction $\theta_{\alpha,X,\omega}$ equals to the curvature of visible part divided by the total curvature of the object's surface. Therefore, $\theta_{\alpha,X,\omega} = \frac{C(V_\alpha(X) \cap \omega)}{C(\omega)}$, $V_\alpha(X) = \bigcup_{x \in X} V_\alpha(x)$. Of course, we suppose here that all the objects in Ω have nonzero curvatures.

Once we have determined which parts of every object are shown in our set of photos X, the quality equation re-

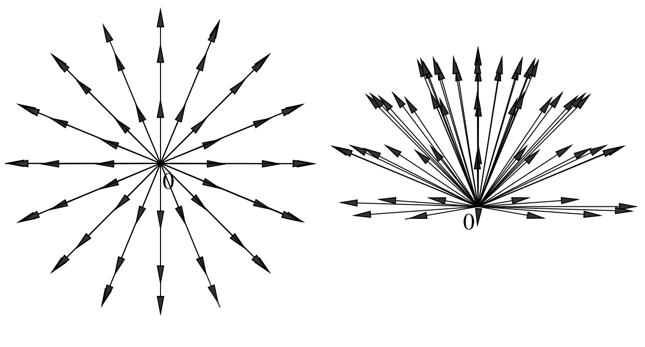

Fig. 9. The set of 65 predefined view directions

mains the same:

$$I_\alpha(X \subseteq S \otimes D) = \sum_{\omega \in \Omega} q(\omega) \cdot \frac{\rho_\omega + 1}{\rho_\omega + \theta_{\alpha,X,\omega}} \theta_{\alpha,X,\omega}.$$

3.2 Choosing views

As we have mentioned before, a set of views, giving a good representation of a scene, could be obtained by a greedy search. The greediness means choosing a best viewpoint at each step of the algorithm. Starting from the best viewpoint, at each step we find the best viewpoint for the unexplored part of the scene. The algorithm stops when the quality of the set surpasses the desirable level of the comprehension.

More strictly: having given a threshold $0 \le \tau \le 1$ (the desirable level of the comprehension), one should find a set of viewpoints $M_k \subseteq S \otimes D$ such as $\frac{I_\alpha(M_k)}{I_\alpha(S \otimes D)} \ge \tau$. Here $I_\alpha(M_k)$ means observation quality for the set of photos M_k, and $I_\alpha(S \otimes D)$ is the observation quality for the set of all possible photos.

At the beginning $M_0 = \emptyset$, each step i of the algorithm adds to the set the best view (s_i, d_i). $M_i = M_{i-1} \cup \{(s_i, d_i)\}$:

$$I_\alpha(M_{i-1} \cup \{(s_i, d_i)\}) = \max_{x \in S \otimes D} Q(M_{i-1} \cup \{x\}).$$

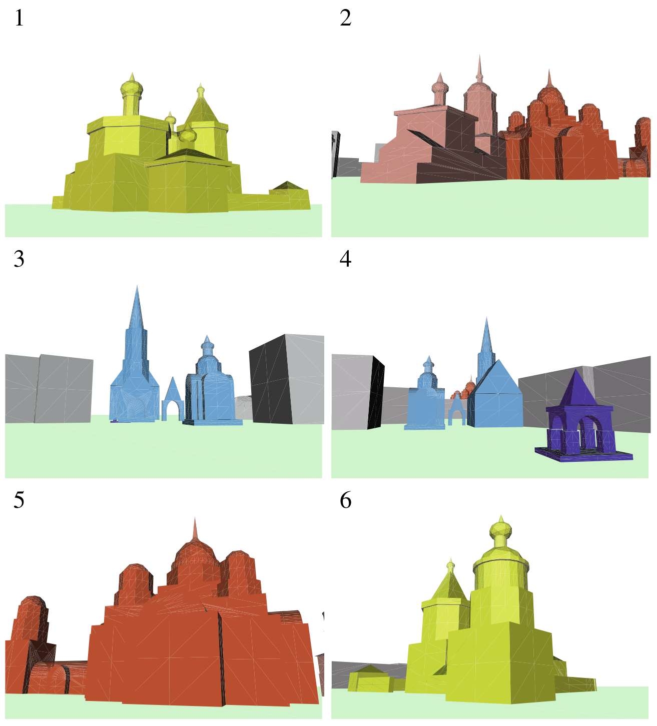

Fig. 10. The six views are shown in the order they were selected by the greedy search algorithm

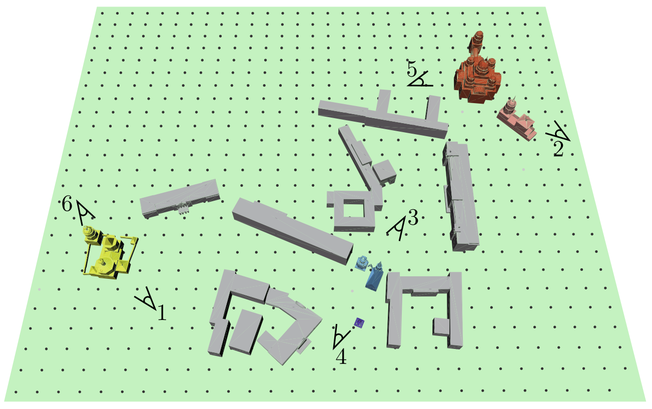

Fig. 11. The virtual town map. The dark dots represent the set of viewpoints. The viewpoints are numbered in order they were selected by the greedy search algorithm. Refer to Fig. 10 for the "photos"

Figure 10 shows a set of "photos" of a virtual town model made by the algorithm. A map of the town is shown at Fig. 11. The viewpoints are indicated on the map too.

In this scene each building represents a single object. The importance contribution $q(\omega)$ is taken equal to 10 for the churches and the chapels, and equal to 2 for the ordinary buildings. Predictability parameter ρ_ω is equal to 1 for the churches and chapels and equal to 0.1 for other buildings. The stopping condition τ was taken equal to 0.95 and the view angle is equal to $\frac{\pi}{3}$.

4 Dynamic exploration

A set of images is a good way to represent a simple scene. But sometimes it is not very easy to understand a scene if the set of viewpoints does not supply information on how one can walk from one point to another. Refer to Fig. 10 for an example. The scene is not complex, but the set of photos confuses the user. In such a case, the best solution is to give the user the ability to view a film made by a virtual camera, a film that gives the user a general knowledge of a scene.

The goal of this section is to develop a local exploration technique allowing continuous exploration of the virtual world. In Sect. 2.2 we cited works of Plemenos for aesthetic criteria of the film quality. Here we add two rules:

1. The movie should not be very long, but it must show as much information as possible.
2. Moreover, there is also the problem of collision detection – the camera should not pass through objects.

The main idea of this section is to take the set of views, calculated with the above methods, and then to join the views with the camera path.

4.1 Camera trajectory – collision detection

In order to avoid collisions, we compute a displacement map: an analytic visibility graph $\hat{G} = (S, \hat{E})$ is to be calculated, where an arc $(s_1, s_2) \in \hat{E}$ if and only if there are no obstacles between the viewpoints $s_1 \in S$ and $s_2 \in S$.

Thus, we define the camera trajectory as a broken line between viewpoints. Later we show how to smooth the trajectory obtained.

The graph $\hat{G}$ can be quickly computed using a slight adaptation of the point-to-point visibility calculation algorithm, introduced in [20]. Note that it is also used in the previous section. The visibility is to be computed between two viewpoints instead of visibility between viewpoints and vertices of a scene. All triangles that do not intersect the viewpoint plane are to be eliminated. The rest is to be projected as it is shown at Fig. 12.

Then the weight function $w(e) : \hat{E} \to \mathbb{R}^+$ is to be specified for each arc of $\hat{E}$. For example, the Euclidean distance could be considered as such a weight function. Then if there are two arbitrary viewpoints a and b and there is a linked component of the graph which contains both a and b, the shortest path between the viewpoints can be found using the Dijkstra's algorithm. There will be no collision between the broken line (the shortest path) $P = (a = p_1, p_2, \dots, p_n = b)$ and objects of the scene due to consequent visibility between viewpoints p_i and p_{i+1}, $1 \le i \le n-1$.

4.2 How to take into account camera direction

Now we can choose paths with respect to obstacles. The next question is: if there are two viewpoints and the camera has to move from one to another, which path is to be chosen? A naive answer is to connect the viewpoints with the shortest path (where the weight function $w(e) : \hat{E} \to \mathbb{R}^+$ is the Euclidean distance). But it does not guarantee that the path consists of good viewpoints. This

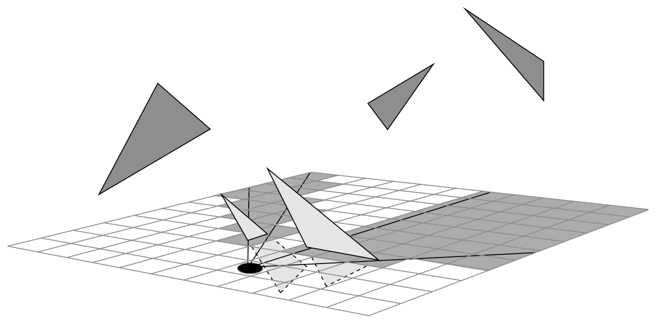

Fig. 12. The rasterized plane represents a set of viewpoints to be processed. The *dark triangles* are eliminated due to *Z*-coordinates. The viewpoint marked by the *black circle* is visible from *white pixels* and is not visible from the *gray ones*

drawback is serious, and, in order to avoid it, we have to introduce additional costs and discounts.

Let us suppose that there are two views (s_1, d_1) and (s_2, d_2), without obstacles between them, $(s_1, s_2) \in \hat{E}$. Let us consider factors influencing the quality of a film. We suppose that the resulting weight function is the superposition of different costs and discounts.

– First of all, one must take into account the cost of moving the camera from one point to another (proportional to the Euclidian distance):

$$c_m((s_1, d_1), (s_2, d_2)) = \|\boldsymbol{s_2} - \boldsymbol{s_1}\|.$$

– The cost of turning the camera is to be considered also (proportional to the rotation angle):

$$c_t((s_1, d_1), (s_2, d_2)) = \frac{\boldsymbol{d_1} \cdot \boldsymbol{d_2}}{\|\boldsymbol{d_1}\| \cdot \|\boldsymbol{d_2}\|}.$$

– Now let us introduce the discount c_q that forces the camera to pass via "good" viewpoints. The following empiric formula augments displacement costs between "bad" viewpoints and reduces near "good" ones.

$$c_q((s_1, d_1), (s_2, d_2)) = 1 - \frac{I_\alpha(s_1, d_1) + I_\alpha(s_2, d_2)}{2 \max\limits_{x \in S \otimes D} I_\alpha(x)}.$$

Figure 13 shows an illustration for the costs. There are three different paths from the left viewpoint to the right one. Paths 1 and 3 have the same length and their intermediate points show equal information (the church is symmetric). But one can conclude that 3 is worse, because it requires more efforts for camera turns. Path 2 is shorter than 1, but in spite of this 1 is better, because its intermediate viewpoint shows the church entirely, while only a small part of the wall is visible from the intermediate viewpoint of Path 2. Thus, the first path is the best one.

Since the aesthetic criteria force to take into account camera directions, the graph $\hat{G}$ is to be extended. Let us redefine the graph as follows: $\hat{G} = (S \otimes D, \hat{E})$, where an arc $((s_1, d_1), (s_2, d_2)) \in \hat{E}$ if and only if there are no obstacles between the viewpoints $s_1 \in S$ and $s_2 \in S$. Note that for two viewpoints s_1 and s_2 we can have 130 photos (the set D consists of 65 predefined view directions). If there is no obstacles between the viewpoints s_1 and s_2, then we add 130 arcs of type $((s_1, d_x), (s_2, d_y))$ to the set $\hat{E}$.

The weight function for an arc between two views could be expressed as follows:

$$\begin{aligned} w((s_1, d_1), (s_2, d_2)) = {} & \alpha_m \cdot c_m((s_1, d_1), (s_2, d_2)) \\ & + \alpha_t \cdot c_t((s_1, d_1), (s_2, d_2)) \\ & + \alpha_q \cdot c_q((s_1, d_1), (s_2, d_2)), \end{aligned}$$

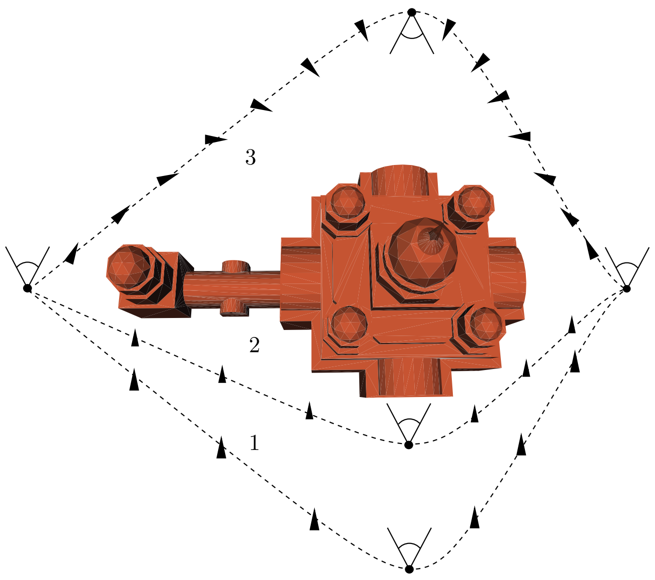

Fig. 13. Three different paths from the left viewpoint to the right one are shown by *dashed lines*. Paths 1 and 3 have the same length, but 3 is worse, because it requires more efforts for camera turns. Path 2 is shorter than 1, but in spite of this 1 is better, because its intermediate viewpoint shows the church entirely, while only a small part of the wall is visible from the intermediate viewpoint of path 2

where α_m, α_t and α_q are the coefficients of relative importance of the different costs. For example, if the wage of viewpoint quality is more important than the cost of camera displacement, α_q is to be chosen greater than α_m. Having defined the weight function, one obtains a trajectory between two views by the Dijkstra's algorithm.

4.3 Constructing the camera trajectory

Now let us remember the set of photos from the previous chapter. So, one has found the set $M_k \subseteq S \otimes D$ of views, and the trajectory of the camera is to be determined. As we have mentioned before, the objective is to connect the views with the camera path.

We solve the problem by converting it to the travelling salesman problem (TSP). The control set of photos is the set of cities to visit. The travelling costs between pairs of views are to be determined as the length of the shortest paths between them in the graph $\hat{G}$.

The resulting trajectory could be computed as the shortest Hamiltonian path (or circuit, if we would like to return the camera to initial point). Note that this instance of TSP is natural and satisfies the triangle inequality constraint. That is, c_q breaks the triangle inequality in the graph $\hat{G}$, but the TSP instance satisfies it. Indeed, the travelling costs are defined as lengths of *shortest* paths in $\hat{G}$, thus, for any two cities A and B there is no circuit shorter than the shortest path between A and B. Unfortunately, the TSP problem is NP-complete even if we require that the cost function satisfies the triangle inequality. But there exist good approximation algorithms to solve the problem. For example, Christofides in [5] has proposed a constant-factor approximation algorithm which always finds a tour of length at most 1.5 times the shortest tour.

A solution of the TSP instance produces a trajectory which traverses all control views. But nothing forces us to produce a single trajectory. Suppose, one should observe a city. Cities contain a lot of touristy places, often they are far from each other. And it is quite convenient to take a subway to reach consequently the sights to be observed. Therefore, we may produce a set of walk-throughs.

One can remove long paths from the final trajectory, but it is more logical to split the initial instance of the TSP. This could be done in different ways. The simplest one is to define a threshold value and to eliminate all edges longer than the threshold. Another way is to consider the travelling costs between the control views as a random variable, and then to use a method of gross error detection. All long paths will be removed. Often it is sufficient to eliminate all paths longer than the mean value plus the quadratic error for the random variable.

Figure 14 shows the final set of exploration trajectories for the virtual town we have already presented. The problem was split into three TSP instances by removing all paths longer than the mean value plus the quadratic error.

The camera trajectory is the broken line between the viewpoints, the order is determined by the TSP. Note also four additional views that are added by the resolving process of the problem. They were found by the Dijkstra's algorithm in order to avoid collisions with objects. Figure 15 shows the "photos". For example, there is an obstacle between viewpoints 3 and 4. But there are no obstacles between points 3–8 and 4–8. Then the broken line 3–8–4 does not hit any scene object. Thus, calculating the short-

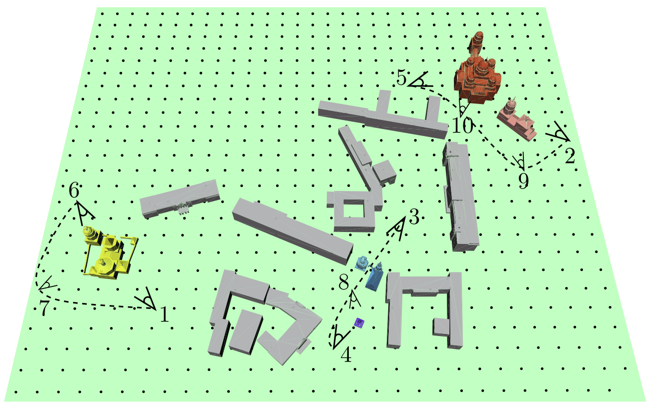

Fig. 14. The viewpoints 7–10 indicate additional "linking" views, the *dashed lines* show three camera trajectories. The viewpoints are numbered in order they were found. Refer to Figs. 10 and 15 for the "photos"

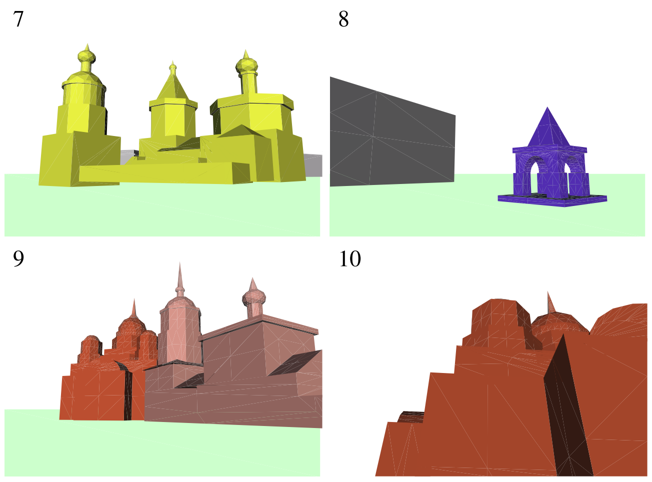

Fig. 15. Four additional "linking" viewpoints were added during solving the TSP. Refer to Fig. 14 for a map of the virtual town

est path in the graph $\hat{G}$ makes the archway accessible during traversal from the point 3 to 4.

Having computed a broken line as the camera trajectory, we are sure that there is no collisions. However, abrupt change of direction in vertices of the line may be distracting. The simple solution is to calculate a spline with the line vertices as control points. For example, the paths in the Fig. 14 are smoothed. But since the spline moves a bit from the straight line, the camera could hit a scene object. Thus, having the spline calculated, one should verify for collisions between the smoothed trajectory and the scene. If somewhere there is a collision, then this part of the spline is to be replaced by the original straight line.

5 Semantic distance

5.1 Structureless vs. semantic

As we have mentioned above, it would be very helpful if a semantic 3D model could be used instead of structureless collection of polygons. Let us consider an example, refer to Fig. 16.

Figure 16a shows a scene with two churches. There are three manually fixed viewpoints. The dashed line is the trajectory, obtained by the above method of local exploration. The exploration is started at the viewpoint 1. It is easy to see that the camera turns aside from the big church in the viewpoint 2 and then reverts back again. The trajectory is very short and smooth, but the exploration is confusing for the user.

Figure 16b shows another trajectory, where the exploration starts in the viewpoint 1, passes via 3 and terminates in the point 2. The camera turns aside from the big church only when its exploration is complete. Despite the increased length of the trajectory, the exploration is more clear.

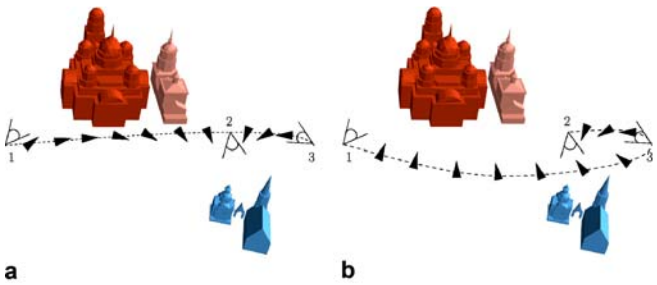

Fig. 16a,b. Two different trajectories. **a** During traversal from the point 1 to 3 the camera turns aside from the big church and then reverts back again. **b** The camera turns aside from the big church only when its exploration is complete

Thus, we postulate that the above method of local exploration could be improved by regrouping the control photos in dependence on *relativeness* of information they show. In other words, if two photos show the same object, they are to be consequent.

It is natural as a concept, but it is necessary to measure relativeness of information shown on two control photos. Remember that we have a scene along with its disjunction into a set of objects. As a "photo" we refer a pair (s, d) of viewpoint and view direction. Therefore, we always know what objects are shown at a given "photograph", and there is not any kind of object recognition.

The relativeness (or semantic distance) can be defined in many ways. For example, Foo et al. in [9] define a semantic distance in conceptual graphs based on Sowa's definition. Zhong et al. in [25] propose an algorithm to measure the similarity between two conceptual graphs.

In our work we use semantic networks. First of all, in the semantic network to be processed we create nodes for all the objects of the scene. Then, if some additional information could be provided, auxiliary nodes and relationships are to be added.

An example of such a network is given at Fig. 17. The figure shows a semantic network for a small virtual museum, which contains paintings by Van Gogh ($\mathbf{vG_{p1}}$, $\mathbf{vG_{p2}}$), Monet ($\mathbf{M_p}$), Da Vinci ($\mathbf{dV_p}$) and two sculptures by Da Vinci ($\mathbf{dV_s}$) and unknown sculptor ($\mathbf{Unk_s}$). Importance of relationships is denoted by thickness of lines. Thus, the information that $\mathbf{M_p}$ is created by Claude Monet is more important than the information that $\mathbf{M_p}$ is a painting.

5.2 Distance between two objects in the semantic network

In order to measure semantic distance between two concepts s and t in the semantic network, it is necessary to measure how many relationships are between two objects s and t. The key idea of this section is to transform the semantic network to an undirected flow network and to calculate the maximum flow that could be pushed from the source s to the target t.

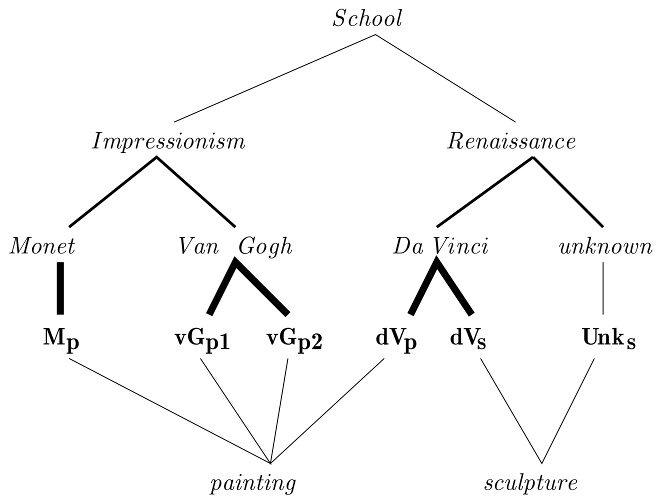

Fig. 17. The semantic network example for a virtual museum. The importance of relationships is denoted by thickness of *lines*

This section is organized as follows: first, we introduce a function to measure distance between *two concepts* in a semantic network. Then we check all necessary conditions and show that the function is the metric. The next section introduces a metric between two *sets of concepts*.

5.2.1 Formal definition of the metric between two concepts

Therefore, we transform the semantic network to an undirected flow network $G = (\mathcal{C}, E_{\mathcal{C}})$. Edges $E_{\mathcal{C}}$ correspond to relationships between concepts $\mathcal{C}$ of the semantic network, capacities $b(i, j) > 0\ \forall(i, j) \in E_{\mathcal{C}}$ are to be set according to the relationship importance. s and t are the source and the sink with unbounded supply and demand, respectively. Figure 18 shows the above network with the associated capacities.

Let us denote the capacity of a xy-cut ($x, y \in \mathcal{C}$) as $\mathrm{cut}(x, y)$ and the capacity of xy-mincut as $\min\mathrm{cut}(x, y)$. Then the distance between two concepts of the semantic network can be defined as follows:

$$D(s, t) = \frac{1}{\min\mathrm{cut}(s, t)}. \tag{4}$$

For continuity reasons we define $D(x, x) \equiv 0\ \forall x \in \mathcal{C}$.

5.2.2 Let us check that D is the metric

Further we show that Eq. 4 defines a metric $D : \mathcal{C} \times \mathcal{C} \longrightarrow \mathbb{R}^+$. Indeed,

1. Capacities are non-negative, $D(x, y) \geq 0\ \forall x, y \in \mathcal{C}$,
2. The identity of indiscernibles is satisfied: $D(x, y) = 0$ if and only if $x = y$,
3. The network is undirected, the symmetry is satisfied, $D(x, y) = D(y, x)$,
4. The triangle inequality is satisfied also: $D(a, b) \leq D(b, c) + D(c, a)$.

Let us suppose that the triangle inequality is not satisfied, then there exists a network and three nodes a, b, c such as $D(a, b) > D(b, c) + D(c, a)$. Then

$$\frac{1}{\min\mathrm{cut}(a, b)} > \frac{1}{\min\mathrm{cut}(b, c)} + \frac{1}{\min\mathrm{cut}(c, a)}. \tag{5}$$

The ab-min cut is also bc-cut or ca-cut. Without loss of generality, let us suppose that ab-mincut is bc-cut. Then $\min\mathrm{cut}(b, c) \leq \min\mathrm{cut}(a, b) \Rightarrow \frac{1}{\min\mathrm{cut}(b,c)} \geq \frac{1}{\min\mathrm{cut}(a,b)}$. Our assumption (refer to inequality Eq. 5) implies a contradiction. Therefore, Eq. 4 defines a metric.

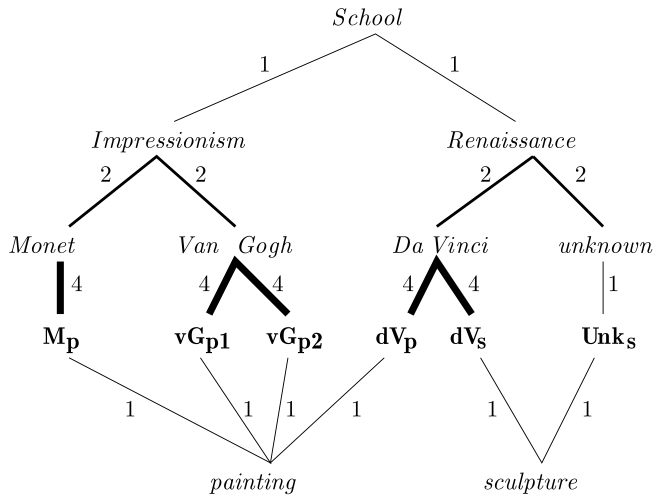

Fig. 18. The semantic network example for a virtual museum. The numbers indicate the capacities for the network $G = (\mathcal{C}, E_{\mathcal{C}})$

5.3 Distance between two sets in the semantic network

Now, having defined the semantic distance between two objects, we can introduce the semantic distance between two "photos". More strictly, if one "photo" shows a set of objects $A \subseteq \mathcal{C}$ and another shows a set of objects $B \subseteq \mathcal{C}$, one can define the similarity distance between the sets as follows:

$$D(A, B) = \frac{\sum_{(a_i,b_j)\in A\times B} D(a_i, b_j)}{|A| \cdot |B|}.$$

The distance between sets is also a metric. Obviously, the non-negativity and the identity of indiscernibles and the symmetry are satisfied. Let us suppose that the triangle inequality is not satisfied, then there exists a network and three subsets A, B, C such as $D(A, B) > D(B, C) + D(C, A)$.

$$\frac{\sum_{A\times B} D(a_i, b_j)}{|A| \cdot |B|} > \frac{\sum_{A\times C} D(a_i, c_k)}{|A| \cdot |C|} + \frac{\sum_{C\times B} D(c_k, b_j)}{|C| \cdot |B|}. \tag{6}$$

$$|A|\cdot|B|\cdot|C|>0$$
$$\Rightarrow |C|\cdot\sum_{A\times B} D(a_i,b_j)$$
$$> |B|\cdot\sum_{A\times C} D(a_i,c_k) + |A|\cdot\sum_{C\times B} D(c_k,b_j).$$

Let us enumerate all the members of the inequality:

$$\sum_{1\le k\le |C|} \overbrace{\sum_{\substack{1\le i\le |A|\\ 1\le j\le |B|}} D(a_i,b_j)}^{k\text{-th member}}$$
$$> \sum_{1\le j\le |B|} \overbrace{\sum_{\substack{1\le i\le |A|\\ 1\le k\le |C|}} D(a_i,c_k)}^{j\text{-th member}}$$
$$+ \sum_{1\le i\le |A|} \overbrace{\sum_{\substack{1\le k\le |C|\\ 1\le j\le |B|}} D(c_k,b_j)}^{i\text{-th member}}$$

For each $(i,j,k)\in[1\dots|A|]\times[1\dots|B|]\times[1\dots|C|]$ there is exactly one inequality $D(a_i,b_j)\le D(a_i,c_k)+D(b_j,c_k)$. But the mapping is also surjective, i.e., all the members are numbered with the triples (i,j,k). Therefore, our assumption (refer to inequality Eq. 6) implies a contradiction. The similarity distance between "photos" satisfies the triangle inequality.

5.4 Example

Now let us show how the metric could be used in scene exploration. In the previous chapter we have defined the travelling costs between pairs of views as the length of the shortest paths between them in the graph $\hat{G}$. Now let us redefine the travelling cost between two "photos" as the length of the shortest path between them *plus the semantic distance between the "photos"*.

Figure 19a presents the shortest exploration trajectory for a small virtual museum we presented above. Refer to Fig. 17 for the semantic network. The exploration starts from the Van Gogh painting ($\mathbf{vG_{p1}}$), then comes Da Vinci painting $\mathbf{dV_p}$, Monet ($\mathbf{M_p}$) etc. The exploration terminates after visiting the second painting of Van Gogh ($\mathbf{vG_{p1}}$).

It is clear that the order is not good. For example, it is better to observe creations of one author consequently. A trajectory, obtained with respect to the semantic network is shown at Fig. 19b. The camera starts with two paintings by Van Gogh ($\mathbf{vG_{p1}}$, $\mathbf{vG_{p2}}$), continues with Monet ($\mathbf{M_p}$), then shows the painting by Da Vinci ($\mathbf{dV_p}$), passes to the sculpture by Da Vinci ($\mathbf{dV_s}$) and terminates with the sculpture by unknown author ($\mathbf{Unk_s}$).

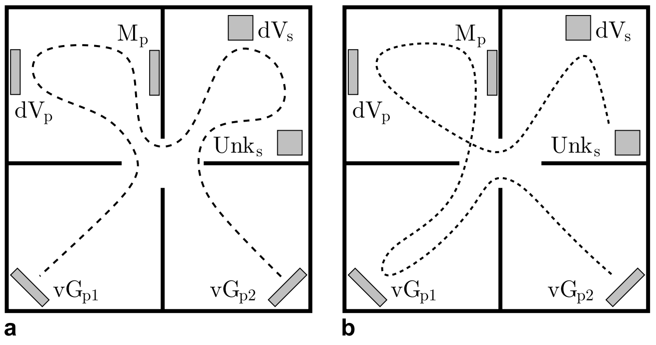

Fig. 19a,b. The virtual museum exploration trajectories. **a** The shortest trajectory. **b** The shortest trajectory with respect to the semantic network shown at Fig. 17

It is easy to see that the second trajectory is more logical than the first one. The trajectory does not interrupt exploration of items by the same author and the camera passes to the renaissance items only when all the impressionists paintings are explored.

6 Conclusion and future work

In this paper we have introduced a non-incremental method of local scene exploration. It allows a camera to navigate inside a model until most of interesting reachable places are visited. The method runs in real-time. In distinction from [6, 24], it guarantees that all interesting places in a scene will be observed. Results produced by the method satisfy the aesthetic criteria defined in Sect. 4. The techniques, presented in this paper, may be easily generalized in order to be used in global exploration too.

A new measure of similarity between objects is also introduced in this paper. It is useful when some additional knowledge of scene structure could be provided. This measure, so called semantic distance, evaluates relationships in the scene and can be used to improve the exploration method.

Semantic networks promise to be a rich area for further research. We are currently defining similarity measure between objects, but it should be possible to extend the definitions taking into account user preferences. It will also be an interesting exercise to use machine learning techniques to take into account implicit user preferences. Different people have different tastes, and artificial intelligence techniques could help to handle some uncertainties. Probably, it would be possible to create for each user a database of preferences to improve exploration of further scenes.

Acknowledgement The authors would like to express their gratitude towards the European Community and the Limousin Region for their financial support of the present work.

References

1. Barral, P., Dorme, G., Plemenos, D.: Visual understanding of a scene by automatic movement of a camera. In: GraphiCon'99. Moscow, Russia (1999)
2. Barral, P., Dorme, G., Plemenos, D.: Scene understanding techniques using a virtual camera. In: Proceedings of Eurographics 2000, Short Presentations, Rendering and Visibility. Interlaken, Switzerland (2000)
3. Blanz, V., Tarr, M.J., Bulthoff, H.H.: What object attributes determine canonical views? Perception **28**, 575–600 (1999)
4. Chazelle, B., Dobkin, D.P., Shouraboura, N., Tal, A.: Strategies for polyhedral surface decomposition: an experimental study. In: SCG '95: Proceedings of the eleventh annual symposium on Computational geometry, pp. 297–305. ACM Press, New York, NY, USA (1995) (DOI http://doi.acm.org/10.1145/220279.220311)
5. Christofides, N.: Worst-case analysis of a new heuristic for the travelling salesman problem. Technical Report 388, Graduate School of Industrial Administration, Carnegie Mellon University (1976)
6. Dorme, G.: Study and implementation of 3D scene understanding techniques. Ph.D. thesis, University of Limoges, France (2001) (In French)
7. Feixas, M.: An information theory framework for the study of the complexity of visibility and radiosity in a scene. Ph.D. thesis, University of Catalonia (2002)
8. Feixas, M., del Acebo, E., Bekaert, P., Sbert, M.: An information theory framework for the analysis of scene complexity. Comput. Graph. Forum **18**(3), 95–106 (1999)
9. Foo, N., Garner, B.J., Rao, A., Tsui, E.: Semantic distance in conceptual graphs, pp. 149–154 (1992)
10. Gooch, B., Reinhard, E., Moulding, C., Shirley, P.: Artistic composition for image creation. In: Proceedings of the 12th Eurographics Workshop on Rendering Techniques, pp. 83–88. Springer, London, UK (2001)
11. Kamada, T., Kawai, S.: A simple method for computing general position in displaying three-dimensional objects. Comput. Vision Graph. Image Process. **41**(1), 43–56 (1988)
12. Lee, C.H., Varshney, A., Jacobs, D.W.: Mesh saliency. ACM Trans. Graph. **24**(3), 659–666 (2005) (DOI http://doi.acm.org/10.1145/1073204.1073244)
13. Noser, H., Renault, O., Thalmann, D., Thalmann, N.M.: Navigation for digital actors based on synthetic vision, memory, and learning. Comput. Graph. **19**(1), 7–19 (1995)
14. Plemenos, D.: A Contribution to the Study and Development of Scene Modelling, Generation and Visualisation Techniques. The MultiFormes project. Professorial dissertation (1991)
15. Plemenos, D.: Declarative modeling by hierarchical decomposition. The actual state of the MultiFormes project. In: GraphiCon'95. Saint Petersbourg (Russia) (1995)
16. Plemenos, D.: Exploring virtual worlds: Current techniques and future issues. In: International Conference GraphiCon'2003. Moscow, Russia (2003)
17. Plemenos, D., Benayada, M.: Intelligent display in scene modelling. new techniques to automatically compute good views. In: GraphiCon'96. Saint Petersburg, Russia (1996)
18. Sbert, M., Feixas, M., Rigau, J., Castro, F., Vázquez, P.P.: Applications of the information theory to computer graphics. In: International Conference 3IA'2002. Limoges, France (2002)
19. Sbert, M., Plemenos, D., Feixas, M., Gonzalez, F.: Viewpoint quality: Measures and applications. In: Computational Aesthetics in Graphics, Visualization and Imaging (2005) (URL http://msi.unilim.fr/basilic/Publications/2005/SPFG05)
20. Sokolov, D., Plemenos, D.: Viewpoint quality and scene understanding. In: Mudge, M., Ryan, N., Scopigno, R. (eds.) VAST 2005: Eurographics Symposium Proceedings, pp. 67–73. Eurographics Association, ISTI-CNR Pisa, Italy (2005)
21. Sokolov, D., Plemenos, D., Tamine, K.: Viewpoint quality and global scene exploration strategies. In: International Conference on Computer Graphics Theory and Applications (GRAPP 2006). Setúbal, Portugal (2006)
22. Vázquez, P.P.: On the selection of good views and its application to computer graphics. Ph.D. thesis, Barcelona, Spain (2003)
23. Vázquez, P.P., Feixas, M., Sbert, M., Heidrich, W.: Viewpoint selection using viewpoint entropy. In: VMV '01: Proceedings of the Vision Modeling and Visualization Conference 2001, pp. 273–280. Aka GmbH (2001)
24. Vázquez, P.P., Sbert, M.: Automatic indoor scene exploration. In: Proceedings of 6th International Conference on Computer Graphics and Artificial Intelligence 3IA'2003, pp. 13–24. Limoges, France (2003)
25. Zhong, J., Zhu, H., Li, J., Yu, Y.: Conceptual graph matching for semantic search. In: Proceedings of the 10th International Conference on Conceptual Structures ICCS 2002, pp. 92–106. Springer, London, UK (2002)
26. Zuckerberger, E., Tal, A., Shlafman, S.: Polyhedral surface decomposition with applications. Comput. Graph. **26**(5), 733–743 (2002)

DMITRY SOKOLOV has been a research fellow at LE2I laboratory at the University of Dijon (France) since 2006. His research area is automatic virtual world exploration.

DIMITRI PLEMENOS is a full professor at the University of Limoges (France). His research area is intelligent techniques in computer graphics, including declarative modelling, intelligent rendering and intelligent virtual world exploration. He is author or co-author of several papers and member of the IPC of many international conferences and journals. Dimitri Plemenos is the organizer and general chair of the 3IA international conference on computer graphics and artificial intelligence.