A Motion-aware Data Transfers Scheduling for Distributed Virtual Walkthrough Applications

J. Pribyl
Dept. of Computer Graphics and Multimedia
Faculty of Information Technology, BUT
612 66, Brno, Czech Republic
pribyl@fit.vutbr.cz

P. Zemcik
Dept. of Computer Graphics and Multimedia
Faculty of Information Technology, BUT
612 66, Brno, Czech Rep.
zemcik@fit.vutbr.cz

T. Burian
Cadwork Informatik CI AG
Cadwork, s.r.o., Czeach team Brno
602 00, Brno, Czech Rep.
burian@cadwork.cz

B. Kudlac
Cadwork Informatik CI AG
Cadwork, s.r.o., Czeach team Brno
602 00, Brno, Czech Rep.
kudlac@cadwork.cz

ABSTRACT

Data transfers scheduling is an important part of almost all distributed virtual walkthrough (DVW) applications. Its main purpose is to preserve data transfer efficiency and render quality during scene exploration. The most limiting factors here are network restrictions such as low bandwidth and high latency. Current scheduling algorithms use multi-resolution data representation, priority determination and data prefetching algorithms to minimize these restrictions. Advanced priority determination and data prefetching methods for DVW applications use mathematic description of motion to predict next position of each individual user. These methods depend on the recent motion of a user so that they can accurately predict only near locations. In the case of sudden but regular changes in user motion direction (road networks) or fast moving user, these algorithms are not sufficient to predict future position with required accuracy and at required distances. In this paper we propose a systematic solution to scheduling of data transfer for DVW applications which uses next location prediction methods to compute download priority or additionally prefetch rendered data in advance. Experiments show that compared to motion functions the proposed scheduling scheme can increase data transfer efficiency and rendered image quality during scene exploration.

Keywords

distributed virtual walkthrough, next location prediction, motion function, Markov chain, prefetching, virtual environments

1 INTRODUCTION

The initial purpose of DVW applications was to realize a virtual tourism task which allows users to visit places of interests without physically entering them (like Google Street View).

Advances in graphic and computing performance of mobile devices, sharp growth in their market and various digital media data archives created commercially or community contributed, further increase potential and usage of DVW applications. Compared to classical desktop computers, content explored within mobile devices can be associated with richer context, like location, weather, traffic, etc.

Applications with high potential in this field are augmented reality tourist guide called LifeClipper, mobile augmented reality application Nokia City Lens, or intelligent navigations such as AIDA. Instead of focusing solely on determining routes to a specified target, the AIDA system utilizes analysis of driver behavior to identify a set of goals the driver would like to achieve (e.g. business or shopping districts, tourist areas, or real-time event information related to traffic).

AIDA visualizes all the data in a 3D scene which can help the driver understand better and interpret the delivered information. Based on the driver's motion, both the visualized information and 3D data for rendering 3D scene are downloaded on demand from a remote server via wireless connection.

The main bottleneck of DVW applications is network connection with restrictions, for example low bandwidth or higher latency, especially on wireless networks, so that transfered data can not be received by clients in time. Scheduling algorithms can reduce the impact of these restrictions with the help of multi-resolution data representation, priority determination and data prefetching algorithms so that they can increase quality of rendered scene and data transfer efficiency.

Current scheduling algorithms for DVW applications widely use mathematic description of motion (motion functions) to predict next motion of each individual user. The predicted position can be used to compute download priority or to prefetch scene parts in advance. Unfortunately, these methods are accurate only for prediction of near future positions, and their accuracy decreases also in the case of sudden changes in user motion direction. For networks with higher latency, low bandwidth or just for fast moving user a prediction method with higher accuracy enabling to predict farther-positions is needed to keep data transfer efficiency and scene quality as high as possible. From another point of view, for some applications the scene quality can be a much more important parameter than the data transfer efficiency.

In this paper, we propose a systematic solution to scheduling of data transfer for DVW applications which uses next location prediction (NLP) methods which increases both data transfer efficiency and quality of rendered scene. Our solution is based on two key insights. First, NLP methods have much higher prediction accuracy compared to motion functions. Second, NLP methods can be adaptively constructed according to the multi-resolution data representation. This feature allows the scheduling algorithm to prefetch missing data at specified resolution as is required by a rendering algorithm.

2 RELATED WORK

This section briefly introduces state-of-the-art scheduling mechanisms for DVW applications.

2.1 Visibility determination

Scheduling methods for DVW applications widely use area of interest (AOI) determination algorithms [sch96], [hes98], [chi98], [li04]. Instead of downloading complete scene, it is suficient to transfer only data in spherical area around an observer. Objects inside this area can be regarded as objects from potentially visible set (PVS) with high download priority. Wang et al.[wan09] additionally divide AOI to sections with different download priority taking into account view frustum and distance from the observer. The AOI based scheduling methods are not suitable for more complex scenes such as terrains. Marvie et al. [mar11] use PVS to schedule data transfers for complex virtual scene divided to cells by a regular grid. Download priority of PVS of adjacent cells is determined by simple ray-casting method based on last two viewpoints. The visibility determination is also used to eliminate transferring scene parts not visible to an observer.

2.2 Motion function

Scheduling algorithms based on motion functions use vector representation of object motion, position and direction. Motion functions can be classified into linear and nonlinear [tao04], which are more accurate than the linear methods. Chim [chi98] proposes exponential weighted moving average (EWMA) motion prediction scheme which assigns different weights to past motion vectors where more recent vectors have higher weights. CyberWalk [chi03] use the EWMA scheme to achieve at least a minimum resolution of the scene. Scheduling algorithm proposed in [tel01] selects objects to be sent to client device based on integral of a benefit measure along predicted path. The prediction is made at server and is based on the assumption that once a particular type of motion is started, it will continue in the near future. This approach does not consider any previous positions. A motion-aware approach which uses state-of-the-art recursive motion function [tao04] for efficient evaluation of continuous queries on 3D object databases is described in [ali10]. The predicted positions here are used to determine download priorities of progressively recorded objects inside a virtual scene so that only exact portion of each object will be downloaded based on the computed priorities. In [sch06] an algorithm for speculative prefetching of terrain tiles is presented. It predicts viewpoint motion by fitting a spline through a list of last positions so the tiles that are predicted to become visible in the near future can be prefetched in advance.

2.3 Next location prediction

Next location prediction (NLP) methods make the assumption that there is a certain regularity in the movement patterns so they are not completely random [gon08]. Only in [lau08] are mentioned advantages of NLP method for virtual environments. Here, a hybrid method, where a combination of a mouse motion prediction and NLP based on statistical approach is used to reduce latency. As the statistics are collected from zone to zone within a scene divided by regular grid, the information about continuous movement is lost.

In [bat02] a simple Markov model is used to estimate transition probabilities between adjacent cells based on movement history database. In [gam12] is used Markov chain of order *m*, which further increases prediction accuracy. Work [ash09] and [gam12] cluster GPS data into frequently visited locations (POI) [zho04]. Clustering to POIs is not suitable for DVW aplications, because granularity of requested prediction for common rendering algorithms is much higher. Mixed Markov-chain model [asa11] has been proposed to model behavior of individual pedestrians as well as a group of pedestrians with similar behavior. It uses combination of Markov chain and Hidden markov model (HMM) to construct the universal predictor. This approach has higher quantity compared to stand-alone Markov chain based methods, but the HMM method has high training complexity.

3 ENVIRONMENT DESCRIPTION

As each data transfer scheduling algorithm is closely related to used rendering algorithm and scene data representation, we will first briefly introduce our experimental framework.

3.1 Multi-resolution data representation

Multi-resolution data representation allows streaming and rendering of scene parts at coarser resolutions in the case of slow network connection, fast moving user or limited rendering capabilities of target devices. In our framework, the multi-resolution data is represented by three data layers including terrain geometry layer, terrain textures layer and cartographic layer (see Figure 1).

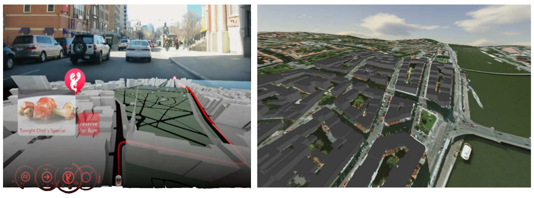

Figure 1: AIDA - 3D visualization and navigation system with augmented reality [aid13] (left), and our streaming and rendering system (right).

3.1.1 Terrain geometry and texture layer

The terrain layer contains height-map tiles (obtained from ASTER global digital elevation model [ast13]) which are further organized into an elevation data pyramid (see Figure 2).

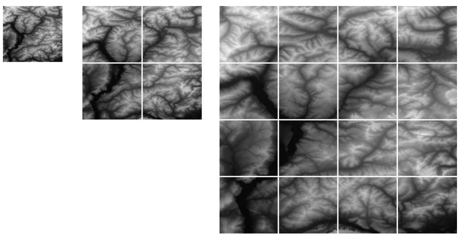

Figure 2: Each tile from coarser level (left) can be covered by its four children (center), continuing recursively (right) to the bottom of the pyramid. Each child tile covers one quarter of the area covered by its parent tile.

Each tile through all levels of the elevation data pyramid has resolution of 128×128 height points. The top level of the pyramid contains single tile which covers the whole terrain at a much coarser resolution. The four child tiles cover the same area as their parent tile thus resulting in double resolution. Tiles at the bottom of the elevation data pyramid cover whole terrain at highest possible resolution.

The texture layer contains high resolution orthographic texture tiles which are mapped onto the terrain tiles. Each tile has resolution of 256×256 pixels and is also included inside a texture data pyramid (see Figure 3) similarly to the terrain tiles.

Figure 3: Part of the textures data pyramid which is created similarly like the elevation data pyramid.

As the elevation and textures data is obtained from real datasets, it is defined, that each elevation tile is covered by an array of 4×4 texture tiles. Consequently, each 128×128 terrain tile is covered by a texture data with resolution of 1024×1024 pixels.

3.1.2 Cartographic layer

The cartographic layer is created from Open Street Map (OSM) cartographic database, which contains definition of streets (geographic location, names, types, etc.), buildings (outlines, nested outlines, roof types, height, floor levels etc.) and other information. Each cartographic tile covers a single terrain tile at its finest resolution. No level of detail for the cartographic data is used. Download priority defined between the three layers is application dependent and is not the main point of interest of this paper.

3.2 Rendering algorithm

The scene is rendered using a set of concentric square rings around the user. Each ring is composed of a constant size grid of small patches. As the user moves, the patches which fall outside a render ring are asynchronously updated with new data from the three data layers at appropriate resolution (see Figure 4). To prepare and render each patch, only small parts of one or more terrain, texture and cartographic tiles are needed.

In our experimental evaluation we set that each ring is composed of a 12×12 grid of patches. Each patch covers an array of 13×13 elevation points. Consequently, each ring (whole square) needs at most 4 terrain tiles, 36 texture tiles and 4 cartographic tiles to prepare all its patches for rendering. As the user continuously

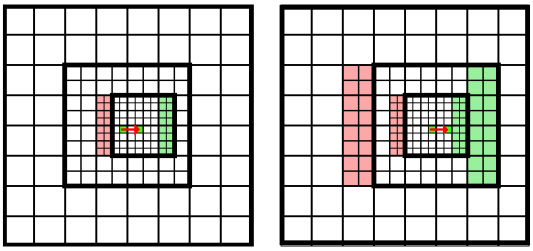

Figure 4: Example of ring patches update for three concentric rings. The red arrow symbol represent a moving user. The red patches have to be updated with the data covered by the green patches at particular resolutions.

moves, only subset of data tiles which cover the updated patches is needed.

Patches outside view frustum are not rendered but data tiles needed by these not visible patches are scheduled to be downloaded with low priority. This behavior is application dependent.

4 MOVEMENT DESCRIPTION

The movement of each user is defined as a continuous sequence of geographic coordinates (gps trajectory). Instead of working directly with gps trajectories, the NLP methods use mainly sequences of places of interest (places with high density of visiting users or places where the users stay for a long time etc.). These places can be discovered from input trajectories using various clustering mechanisms [ash03].

Once the clustering is finished, the input trajectories are encoded into sequences of visited places of interest (e.g. *home* $\rightarrow$ *work* $\rightarrow$ *shop* $\rightarrow$ *home*). Unfortunately, this approach is not very suitable for streaming of 3D virtual environments, because granularity of prediction based on the visited places of interests is usually too high to fit requirements of common streaming and rendering algorithms.

4.1 Trajectories projection

Instead of finding individual places of interest, we virtually divide the whole environment into a regular grid of square cells and project all the input GPS trajectories according to it so that each trajectory can be described as a continuous sequence of adjacent cells. The projection is repeated several times with a different resolution of the grid (see Figure 5).

The idea of the multiple resolutions of the grid is that particular render rings can be associated with selected resolution of the grid. It allows the NLP to be performed at particular resolution, thus allowing to compute priority or prefetch data tiles at appropriate resolution (see section 6).

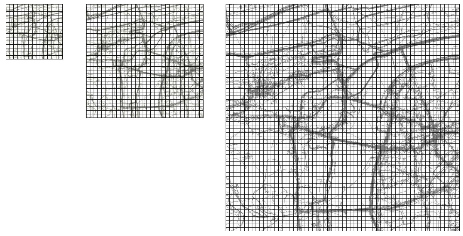

Figure 5: Subset of input trajectories projected to the regular square grid at various resolutions (increased by power of two from left to right).

4.2 Trajectories encoding

Once the projection is finished, each trajectory is encoded as a continuous sequence of adjacent cells (green cells in Figure 6) at selected resolution.

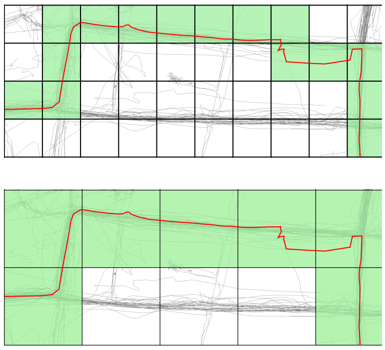

Figure 6: Example trajectory (red) projected at finest resolution (top) and one level coarser resolution (bottom).

Each projected trajectory is further encoded by chain code of eight directions [fre61] (see Figure 7) to efficient storage, and fast evaluation of prediction queries. For example, the red trajectory starting from left in Figure 6, will be encoded as a sequence of directions $2 \rightarrow 0 \rightarrow 0 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 4 \rightarrow 2 \rightarrow 2 \rightarrow 4 \rightarrow 4$ at the finer resolution and as a sequence $0 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 2 \rightarrow 4$ at the coarser resolution.

7	0	1
6		2
5	4	3

Figure 7: Codes for eight possible movement directions from current (center) cell.

4.3 Trajectories storage

Trajectories encoded in the form of 8 directions are stored in a form which will be suitable for further described NLP methods. Assume that a whole trajectory is composed of a sequence of adjacent cells at selected resolution. We take successively each cell of the trajectory starting from the first one and determine specified number of next successive movement directions from that cell along the trajectory. Next 9 successive directions from each cell are used (see section 7). Let's assume that each direction can be encoded using 3 bits. Then a sequence of 9 directions can be stored using 27 bits plus 4 bits to encode its length, because sequences of last 9 cells can be shorter. Therefore, each sequence of directions can be stored using single 32 bit integer value. This value forms a local movement pattern defined for each cell of a trajectory. These local movement patterns are computed for each cell for all input trajectories projected at all resolution levels. The local movement patterns form a knowledge database which is used as an input for the following NLP methods.

5 NEXT LOCATION PREDICTION

NLP methods suitable for our case have to satisfy several requirements: efficient learning, fast adaptation to new behavior of each individual user, high prediction accuracy and quantity, and fast evaluation of prediction queries. The most important requirement is prediction accuracy, because each wrong prediction decreases data transfer efficiency and rendered scene quality.

Learning NLP methods can be considered efficient, if the knowledge database can be modified by adding or removing trajectories so the prediction methods are not forced to be completely relearned. Considering fundamental characteristics of NLP methods [pet06], this requirement is met by both the Markov chain based predictor [ash03], [pet06], [gam12] and also by the K-state predictor [pet03].

5.1 Markov chain based predictor

The Markov chain (MCH) based predictor is based on the definition of Markov chain of order j . A Markov chain of order j selects its next state depending upon j past states. In our case, its state space is defined by the eight movement directions. Each Markov chain operates with transition frequency value which counts an overall number of applications of corresponding movement direction after a sequence of j past movement directions (see Figure 8).

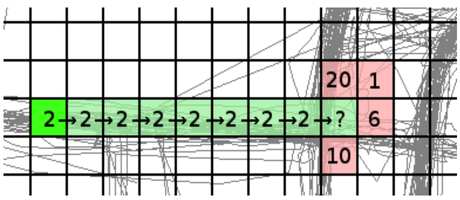

Figure 8: Local movement pattern with length of nine directions related to the left green cell.

The local movement pattern shown in Figure 8 moves the user from the green cell to current cell marked as "?" by application of 8 right transitions ($2 \rightarrow$). The 9th transition from the pattern moves the user to one of the red cells. The red cells contain transition frequency counters incremented by trajectories with the same local movement pattern related to the leftmost green cell.

The cell with highest value of the frequency counter can be selected as a prediction result. Confidence of such prediction is computed as the ratio between the value of selected frequency counter and the sum of all frequency values adjacent to cell "?". If confidence of the prediction is less than 90%, it will be marked as not confident. The lower the confidence threshold, the lower the accuracy. If the confidence computation is skipped, the prediction accuracy is decreased by 5-8% depending on prediction resolution level. We decided to set the confidence threshold to 90% because the prediction accuracy is the most important parameter for us. If the confidence is for example 50%, i.e. two directions should be considered as a prediction result, then it is not reasonable to apply the prediction result.

MCH based predictor has a property that is cannot fast adapt to changes in habits [pet06] neither temporary behavior changes (street closures because of road works etc.) of individual users. Consequently, this characteristic can lead to a confident but a wrong prediction for quite long time until the frequency counters reflect some change.

5.2 K-state predictor

The concept of k-state predictor (KSP) as a NLP method inside a smart office building has been introduced in [pet03]. It can be also succesfully used with the trajectories encoded by chain code of eight directions.

The KSP is constructed as a simple finite automaton with k-states. Currently, we use only a 2-state predictor (2SP) with eight contexts where each context represents one direction. The two states are a weak and a strong state (see Figure 9).

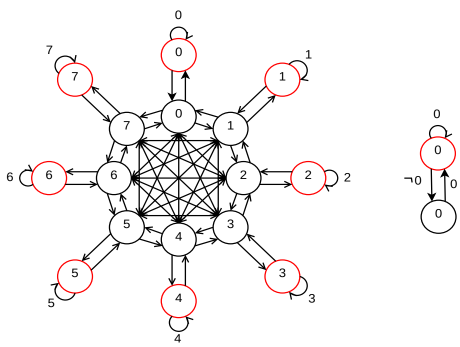

Figure 9: The 2SP with eight strong (red) and eight week states. Each context of the predictor represent one direction from the used chain code of eight directions.

In case the 2SP is in the strong state then the appropriate movement direction is returned as prediction result, otherwise no prediction is returned. Instead of increment frequency counters, the 2SP switches between its contexts and states. Each 2SP starts with undefined context. After applying e.g. eight right directions from the local movement pattern shown in Figure 8 followed by single up direction, it will switch to up context at the weak state. If the same pattern is repeated followed by the up direction again, the up context will switch to the strong state and the up direction will be returned as a prediction result. This functionality can be easily extended by other dimensions like time, day of a week etc.

5.3 Proposed prediction scheme

Input of the prediction scheme is a current movement pattern. It is described by a reference cell and sequence of 8 movement directions. Current cell of the user can be determined by applying these 8 directions from the reference cell. All local movement patterns (stored at client) related to the input reference cell are loaded and only the patterns which match the current movement pattern are selected. As the current input movement pattern has a length of 8 directions and the stored movement patterns have a length of 9 directions, the 9th direction can be used to construct the 2SP.

If the 2SP does not return any prediction, the MCH based predictor will be used. As the MCH based predictor needs the knowledge database of all users in the system, it will be performed and evaluated at server. The 9th direction from all the matching local movement patterns (stored at server) related to the input reference cell are used to increment the frequency counters.

Prediction quantity of both predictors is low because of lack of movement data for new users, users with new behavior or low confidence of the predicted directions. The standard solution to this problem is prediction by partial matching (PPM). We further extend it with application of recursive motion function.

Prediction by partial matching:

Prediction by partial matching (PPM) is based on shortening length j of the local movement patterns so that it successively moves the reference cell toward the current cell until a prediction succeeds or other conditions are reached. The disadvantage of this approach is that the shorter the local movement pattern is, the less prediction accuracy there is. We determine minimum acceptable local movement pattern length so the next location predictors have increased quantity, but still high prediction accuracy. If a predicted cell is based on a local movement pattern shorter than the observed minimal length, we mark such prediction as not confident. Even if such a prediction is not confident, it is still more accurate than motion function based predictors (see section 7). Therefore, such a not confident prediction is used only to compute priorities of missing scene parts, but not for prefetching them.

Recursive motion function:

In case both the Markov chain and 2-state prediction methods did not respond to a prediction query, the proposed prediction scheme predicts next movement using a state-of-the-art recursive motion function (RMF). The input of the RMF method is a sequence of past GPS coordinates so that it can predict next position based on motion function determined from these past positions. The predicted position is clustered using the grid-clustering method only at the finest resolution. In case the cell determined from the predicted position does not differ from the current cell, the RMF predictor is applied again to predict more steps ahead until a difference between the current cell and the predicted cell appears. The RMF method has lower accuracy but high quantity compared to the NLP methods. Therefore, we use the RMF method result only to compute priorities of missing scene parts, but not for data prefetching.

6 DATA TRANSFER SCHEDULING

The main goal of scheduling of data transfer for our DVW application is to achieve effective data transfer with maximum rendering quality during scene exploration. We use a hybrid client-server communication approach, where both client and server can prefetch or request missing scene parts.

6.1 Rendering requirements

Rendering algorithm described in section 3.2 exactly determines data tiles which are needed to render the current view. As the user continuously moves through the scene, the rendering algorithm generates requests to download tiles which are not available in cache memory. The scheduling algorithm first requests tiles for coarser resolution rings, continuing to the finer resolutions. Therefore, the correspondence between the rendered rings and the necessary tiles for the three data layers are determined by outer boundary of each ring.

6.2 Application of the prediction scheme

The scheduling algorithm returns predicted cell which can be used to compute download priority of missing data tiles needed for rendering the current view or to prefetch data tiles needed for rendering the future views.

6.3 Download priority determination

For a fast moving user or slow network connection it is not possible to transfer all requested tiles on time. Even worse, some tiles required for rendering the current view will be downloaded late, so they are no longer needed for rendering the current view.

Priority of all missing data tiles within the given render ring is computed based on a cell C. The cell C is computed as $C = C_c + k * (C_p - C_c)$, where C_c is the center cell of given render ring, C_p is predicted cell and k is a constant which translates the cell C outside the given render ring boundaries. The priority of all missing data tiles is computed as a distance between each missing data tile from the cell C. The less the distance is, the higher the priority.

6.4 Data prefetching

Let's assume that current location of a user is determined by the center cell C_c at selected resolution. The predicted cell C_p is always adjacent to the cell C_c. Every time, the proposed prediction scheme returns confident prediction, the predicted cell C_p is used to identify render rings borders for that cell C_p. These borders specify all the required data tiles if the user will follow the predicted cell C_p. Actually we map one cell from the projection grid to one texture tile from the texture layer at appropriate resolution (see Figure 10). The same principle is applied to the terrain and cartographic layers.

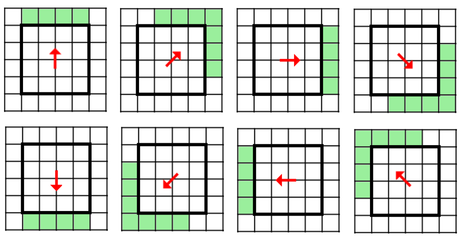

Figure 10: Example one-by-one mapping between tiles from the texture layer and cells of the projection grid. The red arrows sign the predicted direction and the green cells are prefetched data tiles. The black squares are the render rings for the current cell C_c.

Tiles determined by the prefetching algorithm can be scheduled to be downloaded only if no tiles are missing by the current render rings. We select this strategy as we need to achieve both scene quality and data transfer efficiency.

7 EXPERIMENTAL EVALUATION

A proof-of-concept client-server framework which runs on iPad, renders the described virtual environment and exploits the proposed scheduling algorithm was implemented. Prediction accuracy and quantity of the used prediction methods are evaluated as well as render quality and data transfer efficiency during walkthrough the environment.

7.1 Input trajectories dataset

All the experiments were done with trajectories obtained from Open Street Map gpx database from rectangle area specified by two $[latitude, longitude]$ corners as $min = [49.9812545, 14.230042]$ and $max = [50.182172, 14.617306]$ which covers a large city. The dataset contains 2,648 gpx trajectories recorded by various users as continuous sequences of $[latitude, longitude]$ coordinates. The input trajectories are projected to a regular grid at resolution starting from $[0.000278, 0.000278]$ degrees per cell and further increased by the power of two, finishing with five resolution levels.

Accuracy and quantity of the NLP methods are evaluated using 20-fold cross validation (each trajectory set contains 132 trajectories). The validation is performed so that 19 sets are used for learning both the MCH based predictor and also the 2SP. The remaining set is always used to evaluate prediction accuracy and quantity. We repeat this process 20 times for different testing sets and compute the resulting accuracy and quantity by averaging the particular results.

As the trajectories can be obtained only as anonymous records, we cannot assign them to individual users. Instead, we assign all the input trajectories to a single user and evaluate both the MCH based predictor and the 2SP with this assumption. Practically, the quantity of the 2SP will be less than the quantity of MCH based prediction, especially for new users.

7.2 Prediction accuracy and quantity

We decided to store 9 successive directions for each local movement pattern. Then 8 directions can be used to match the current movement pattern with the stored local movement patterns. Experiments with the input trajectories show that 9 directions are sufficient, because the change of accuracy, when the length of the matched sequence of directions is longer than 6 is small (see Figure 11).

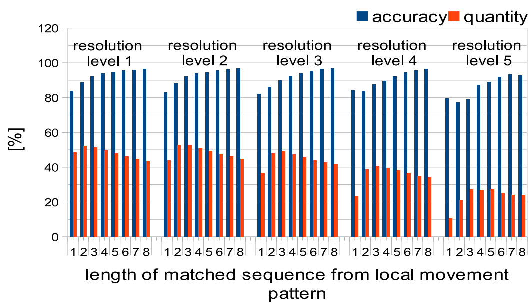

Figure 11: Comparison of accuracy and quantity of MCH predictor for different length of matched sequence of directions and all 5 resolutions of the projection grid.

The prediction quantity is highest for matched sequences of directions with length from two to three direction over all resolution levels as is shown in Figure 11. The quantity is lower for shorter lengths because the predictions are often marked as not confident (confidence threshold is selected at 90%) and also for longer length, because less matched patterns were found. As a compromise between accuracy and quantity we set length of the matched sequence of directions to 6 directions. The accuracy of both the 2SP and MCH based predictor is higher than 90% for all resolutions of the projection grid (see Figure 12).

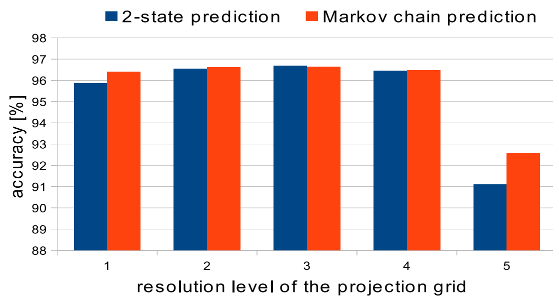

Figure 12: Comparison of prediction accuracy between 2-state (2SP) and Markov chain (MCH) based prediction.

The proposed prediction scheme uses also prediction by partial matching for both MCH based and 2-state predictors. Figure 13 shows effect of PPM to accuracy and quantity of the MCH based predictor for all PPM orders at all resolutions of the projection grid. The PPM order is the maximum allowed length of shortening a sequence of matched directions from the local movement patterns.

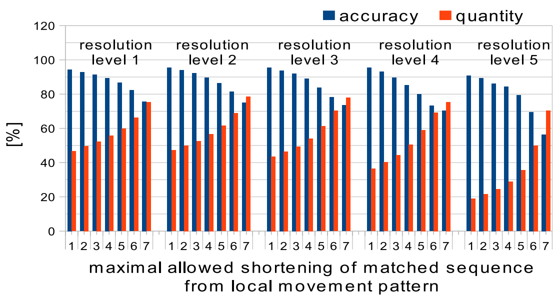

Figure 13: Effect of PPM optimization to accuracy and quantity of MCH based prediction. Initial length of matching sequence is 8 directions.

The results show that the higher the allowed shortening is, the less the accuracy is, but with increased quantity. Based on the PPM results, we have decided to allow the shortening at most by two directions, otherwise the prediction is marked as not confident and is rather used to compute only priorities of current missing data tiles.

Even though prediction accuracy of the highest PPM order at the first resolution level is relatively low, it is still higher than prediction accuracy of the state of the art recursive motion function used as the next location predictor. The prediction accuracy of the recursive motion function is 55% and quantity is 97%. We use 6 past GPS locations to create the RMF predictor and we use it only for prediction at the finest resolution of the projection grid. The accuracy of simple linear predictor is 39% and its quantity is 99%.

7.3 Scheduling scheme evaluation

The goal of the proposed scheduling scheme is to keep both scene quality and data transfer efficiency as high as possible. We measure the scene quality as the ratio between the time a data tile is available for rendering and time the data tile is needed for rendering. The result is computed as weighted average over all tiles for particular levels. Data transfer efficiency is defined as the ratio between amount of downloaded tiles and how these tiles contribute to rendered scene quality.

We changed speed of the user, network latency and connection bandwidth and measured scene quality and data transfer efficiency with the proposed prediction scheme. The experiments are performed on local area network with network latency and bandwidth emulated at linux server using "tc" commands with "netem" kernel component. We use the first set of input trajectories to perform three experiments.

Average data transmission speed on current 3G networks vary from approx. 1000kbps to 3000kbps and latency from approx. 50ms to 100ms [tmc11], but it depends on many conditions. Therefore, in the first experiment (see Figure 14) we set motion speed to 60km/h, bandwidth to 2000kbps and latency to 100ms.

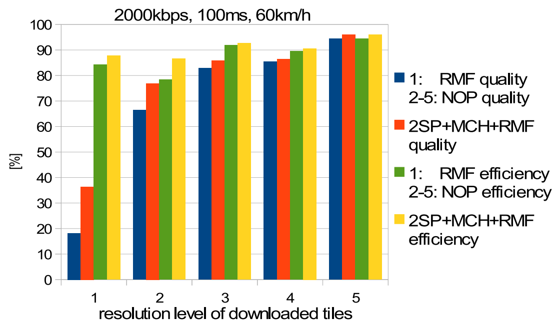

Figure 14: Comparison of quality and data transfer efficiency of the scheduling scheme and RMF (used for the first finest resolution) and no prediction (NOP) for the other resolutions.

The results of the first experiment show that for high bandwidth and relatively high latency for this kind of application the proposed prediction scheme outperforms the RMF used for the finest resolution level and no prediction used for the coarser resolutions. Except the rendering quality at the finest resolution, the results are similar, because the network bandwidth is high so all tiles at all resolutions are downloaded during rendering.

In the second experiment we change speed to 130km/h. The results show (see Figure 15) both increased quality and also significantly increased data transfer efficiency with the proposed prediction scheme at second and third resolution level. It means the downloaded tiles are needed for rendering for longer time compared to the RMF and no prediction approaches. The finest resolution of data tiles was not downloaded at all because the speed is too high for the selected bandwidth and latency.

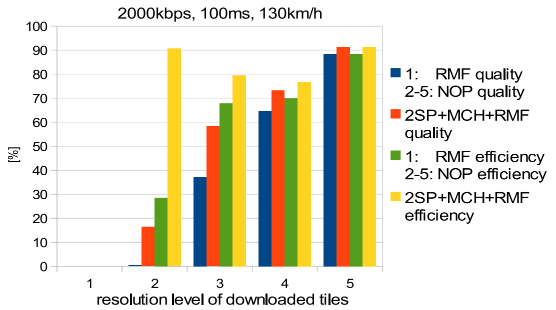

Figure 15: Comparison of scene quality and data transfer efficiency between proposed scheduling scheme and RMF (used for the first finest resolution) and no prediction (NOP) for the other resolutions.

In the last experiment, we set high speed, low bandwidth at 200kbps and zero latency just to show the effect of prediction at low bandwidth networks for fast moving user (see Figure 16).

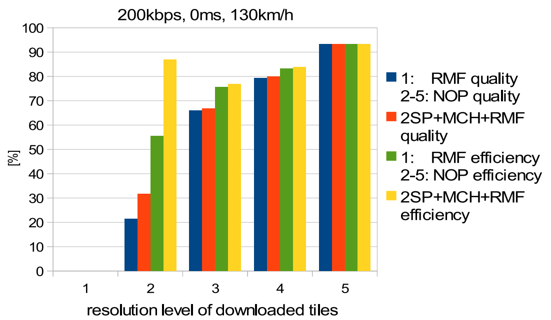

Figure 16: Comparison of scene quality and data transfer efficiency between proposed scheduling scheme and RMF (used for the first finest resolution) and no prediction (NOP) for the other resolutions.

The results show that the proposed scheduling scheme significantly increases data transfer efficiency and scene quality at the second resolution level.

8 CONCLUSION

The proposed scheduling scheme outperforms the RMF at the finest resolution level in both the scene quality and data transfer efficiency. At coarser levels the motion functions cannot be used because of their low accuracy. The effect of the proposed prediction scheme is less significant at the coarser levels, because the amount of time for which the coarser tiles are used for rendering is longer compared to the higher resolution tiles.

For applications, where the transfer efficiency is sufficiently high, it will be possible to predict more than one next location ahead. It will increase scene quality for slow moving users or at high speed networks. With more steps ahead, the possibility that the downloaded tiles will not be used for rendering grows up especially in the case of wrong prediction.

9 ACKNOWLEDGMENTS

This work has been funded by the Technological Agency of Centrum kompetence ve zpracovani vizualnich informaci (V3C - Visual Computing Competence Center, TACR, TE01020415). Special thanks also go to Cadwork Informatik CI AG and Cadwork development team in Brno for supporting of this work.

10 REFERENCES

[aid13] Volkswagen of America, Massachusetts Institute of Technology (SENSEable City Lab and Personal Robots Group of Media Lab). AIDA - Affective, Intelligent Driving Agent. http://senseable.mit.edu/aida/.

[ali10] Ali, M. E.; Tanin, E.; Zhang, R.; aj.: A motion-aware approach for efficient evaluation of continuous queries on 3D object databases. *The VLDB Journal*, Vol. 19, No. 5, 2010, pp.603–632, ISSN 1066-8888.

[asa11] Asahara, A.; Maruyama, K.; Sato, A.; aj.: Pedestrian-movement prediction based on mixed Markov-chain model. In *Proceedings of the 19th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, GIS '11, New York, NY, USA: ACM, 2011, ISBN 978-1-4503-1031-4, pp.25–33.

[ash03] Ashbrook, D.; Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, Vol.7, No.5, pp.275–286, 2003, ISSN 1617-4909.

[ast13] Jet Propulsion Laboratory: ASTER Global Digital Elevation Map. http://asterweb.jpl.nasa.gov/gdem.asp.

[ash09] Ashbrook, D.; Starner, T.: Using GPS to learn significant locations and predict movement across multiple users. *Personal Ubiquitous Comput.*, Vol.7, No. 5, 2003, pp.275–286, ISSN 1617-4909.

[bat02] Bhattacharya, A.; Das, S. K.: LeZi-update: an information-theoretic framework for personal mobility tracking in PCS networks. *Wirel. Netw.*, Vol.8, No.2/3, 2002, pp.121–135, ISSN 1022-0038.

[chi98] Chim, J. H. P.; Green, M.; Lau, R. W. H.; et al.: On caching and prefetching of virtual objects

in distributed virtual environments. In *Proceedings of the sixth ACM international conference on Multimedia*, MULTIMEDIA '98, New York, NY, USA: ACM, 1998, ISBN 0-201-30990-4, pp.171–180.

[chi03] Chim, J.; Lau, R. W. H.; Leong, H. V.; aj.: CyberWalk: A Web-Based Distributed Virtual Walkthrough environment. *IEEE Trans. on Multimedia*, Vol.5, 2003: pp.503–515.

[fal93] Falby, J.; Zyda, M.; Pratt, D. R.; aj.: NPSNET: Hierarchical data structures for real-time three-dimensional visual simulation. *Computers & Graphics* , Vol.17, No.1, pp.65–69, 1993.

[fre61] Freeman, H.: On the encoding of arbitrary geometric configurations. In IEEE Trans. Electron. Comput, pp.260-268, 1961.

[gam12] Gambs, S., Killijian, M.-O., del Prado Cortez, M. N. n.: Next place prediction using mobility Markov chains. In *Proceedings of the First Workshop on Measurement, Privacy, and Mobility*, MPM '12, New York, NY, USA: ACM, 2012, ISBN 978-1-4503-1163-2, pp.3:1–3:6.

[gon08] González, M. C.; R., C. A. H.; Barabási, A.-L.: Understanding individual human mobility patterns. *CoRR*, Nature 453, pp.779-782, 2008.

[hes98] Hesina, G.; Schmalstieg, D.: A Network Architecture for Remote Rendering. In *Proceedings of the Second International Workshop on Distributed Interactive Simulation and Real-Time Applications*, DIS-RT '98, Washington, DC, USA: IEEE Computer Society, 1998, ISBN 0-8186-8594-8, p.88.

[lau08] Lau, R. W.; Chan, A.: Motion in Games. Chapter on Motion Prediction for Online Gaming, Berlin, Heidelberg: Springer-Verlag, 2008, ISBN 978-3-540-89219-9, pp.104–114.

[li04] Li, F. W. B.; Lau, R. W. H.; Kilis, D.: GameOD: an internet based game-on-demand framework. In *Proceedings of the ACM symposium on Virtual reality software and technology*, VRST '04, New York, NY, USA: ACM, 2004, ISBN 1-58113-907-1, pp.129–136.

[mar11] Marvie, J.-E.; Gautron, P.; Lecocq, P.; et al.: Streaming and synchronization of multi-user worlds through HTTP/1.1. In *Proceedings of the 16th International Conference on 3D Web Technology*, Web3D '11, New York, NY, USA: ACM, 2011, ISBN 978-1-4503-0774-1, pp.111–120.

[osm13] Open Street Map. Open Street Map Wiki - Buildings. http://wiki.openstreetmap.org/wiki/Building.

[pet06] Petzold, J., Bagci, F., Trumler, W., and Ungerer, T.: Comparison of different methods for next location prediction. In LNCS, Mobile and Ubiquitous Computing, pp.909-918, 2006.

[pet03] Petzold, J., Bagci, F., et al.: Global and Local State Context Prediction. Artificial Intelligence in Mobile Systems 2003 (AIMS 2003) in Conjunction with the Fifth International Conference on Ubiquitous Computing, Seattle, USA, 2003.

[sch96] Schmalstieg, D.; Gervautz, M.: Demand-Driven Geometry Transmission for Distributed Virtual Environments. *Comput. Graph. Forum*, Vol.15, No. 3, 1996: pp.421–431.

[sch06] Schneider, J.; Westermann, R.: GPU-friendly high-quality terrain rendering. In *The 14th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision 2006 (WSCG 2006)*, Bory, Czech Republic, 2006.

[tao04] Tao, Y.; Faloutsos, C.; Papadias, D.; aj.: Prediction and indexing of moving objects with unknown motion patterns. In *Proceedings of the 2004 ACM SIGMOD int. conference on Management of data*, SIGMOD '04, New York, NY, USA: ACM, 2004, ISBN 1-58113-859-8, pp.611–622.

[tel01] Teler, E.; Lischinski, D.: Streaming of Complex 3D Scenes for Remote Walkthroughs. *Computer Graphics Forum*, Vol. 20, No. 3, 2001, pp.17–25, ISSN 0167-7055.

[tmc11] T-mobile Czech Republic, P3 Communications and BUT: Press release - 3G and 3G networks performance testing results. http://t-press.cz/tiskove_zpravy/2011/1224/.

[wan09] Wang, W.; Jia, J.: An incremental SMLAOI algorithm for progressive downloading large scale WebVR scenes. In *Proceedings of the 14th International Conference on 3D Web Technology*, Web3D '09, New York, NY, USA: ACM, 2009, ISBN 978-1-60558-432-4, pp.55–60.

[zho04] Zhou, C.; Frankowski, D.; Ludford, P.; aj.: Discovering personal gazetteers: an interactive clustering approach. In *Proceedings of the 12th annual ACM international workshop on Geographic information systems*, GIS '04, New York, NY, USA: ACM, 2004, ISBN 1-58113-979-9, pp.266–273.