16th Int. Conf. on Computer Graphics (GRAPHICON), Novosibirsk, Russia, July 2006

Mobile Robot Mapping and Immersive Building Simulation

Rolf Lakaemper[1], Ali M. Malkawi[2], Ravi S. Srinivasan[2], and Longin Jan Latecki[1]

[1]Department of Computer and Information Sciences
Temple University
Philadelphia PA 19122, USA

[2]Department of Architecture
University of Pennsylvania
Philadelphia PA 19104, USA

Abstract

This paper discusses a framework for integrated Augmented Reality (AR) architecture for indoor thermal performance data visualization that utilizes a mobile robot to generate environment maps. It consists of three modules: robot mapping, Computational Fluid Dynamics (CFD) simulation, and AR visualization. The robot mapping module enables the modelling of spatial geometry using a mobile robot. In order to generate steady approximations to scanned 3D datasets, the paper presents a novel "Split and Merge Expectation-Maximization Patch Fitting" (SMEMPF) planar approximation method. The developed SMEMPF method extends the classical Expectation-Maximization (EM) algorithm. It allows for precise adjustment of patches independent from the initial model. The final result is a set of patches identifying planar macro structures that consist of a collection of supported tiles. These patches are utilized to model the spatial geometry under investigation.

The CFD simulation module facilitates the prediction of building performance data based on the spatial data generated using the SMEMPF method. The AR visualization module assists in interactive, immersive visualization of CFD simulation results. Such an integrated AR architecture will facilitate rapid multi-room mobile AR visualizations.

Keywords: *3D Robot Mapping, EM, Augmented Reality, Immersive Building Simulation, CFD.*

1. INTRODUCTION

Augmented Reality is typically used to visualize scientific data, often complex numerical representations plotted in 3D space. It involves the coordination of a number of issues related to AR technology and the type of data to be visualized. Issues related to AR technology include elimination of latency and registration errors, appropriate selection of motion trackers, etc. Data types for AR visualization consist of spatial and scientific data. Spatial data consists of data related to the space (dimensions, inlet / outlet positions, etc) whereas scientific data relies on the type of analyses (indoor thermal, lighting, ventilation performance datasets, etc).

Several research efforts have been established to visualize simulation results in immersive environments such as virtual wind tunnel [1], structural analysis [2], building performance [3,4], etc. CFD simulation is employed to iteratively solve complex heat-mass transfer equations. It involves the ability to evaluate a series of decisions, through setting up initial / boundary conditions, fluid properties, discretization schemes, turbulence models, and approximations. It is extensively used in aerospace, nuclear, automotive, biomedical, environmental, microelectronics, industries, etc. CFD simulation is also performed for building facilities to assess the response of the built environment to specified external conditions. Such simulations aid the design decision-making process for architects and engineers at various stages of building design based on thermal performance. Simulations that use Virtual Reality (VR) or AR environments to visualize and interact with thermal datasets, in addition to HCI, are referred to as *Immersive Building Simulations* [5].

In addition to immersive environments and CFD, Human-Computer Interaction (HCI) technologies such as speech / gesture recognition and eye movement tracking can play a vital role in enabling efficient data manipulation while still being immersed in the environment. Integration of such techniques will aid real-time data interactivity. For buildings, such interactions will facilitate potential applications such as on-site prototyping and diagnostics of Heating, Ventilation, and Air Conditioning (HVAC) system, etc.

Mobile robots have been widely used for generating environment maps [6-8]. The mapping includes simultaneous estimation of the robot position and generating the environment map using sensory input using laser feedback. Such mapping is achieved either by employing a set of robots or a single robot. In the case of multiple robots, partial maps constructed from range sensor data are exchanged with other robots to study shape similarities for overall spatial map creation. On the other hand, a single robot requires multiple runs to generate a similar map. As the robot traverses the space, raw 3D datasets are acquired in real time. A compact environment map composed of few generalized polylines is obtained by converting the raw 3D datasets. Several 3D mapping approaches exist, including Hough transformation, grid-based and EM-based algorithms [9-15]. Although these approaches facilitate such data translations, they build in assumptions such as the number of fitted patches, extent of noise, and/or the order of data points.

This paper discusses a framework for integrated Augmented Reality architecture for indoor thermal performance data visualization that utilizes a mobile robot to generate the spatial data. The mobile AR architecture consists of three modules: robot mapping, CFD simulation, and AR visualization. The robot mapping module enables the modelling of spatial geometry using a mobile robot. It employs a mobile robot to incrementally scan the room using range sensors in real time. To generate steady approximations to acquired 3D data points, the paper presents a novel *Split and Merge Expectation-Maximization Patch Fitting* planar approximation method. The SMEMPF method extends the classical EM algorithm. It allows for precise adjustment of patches, independent from the initial model. The final result is a set of patches identifying planar macro structures (e.g. walls) that consist of a collection of (coplanar) supported tiles. These patches are utilized to model the spatial geometry under investigation. The CFD simulation module facilitates the prediction of building performance data based on the spatial data generated using the SMEMPF method. The AR visualization module assists in interactive, immersive visualization of CFD datasets. Such an

integrated AR architecture will facilitate rapid multi-room mobile AR visualization.

2. INTEGRATED MOBILE AUGMENTED REALITY

The integrated mobile AR for *immersive building simulation* enables interactive, immersive visualization of indoor thermal datasets. It consists of three modules: robot mapping, CFD simulation, and AR visualization, figure 1.

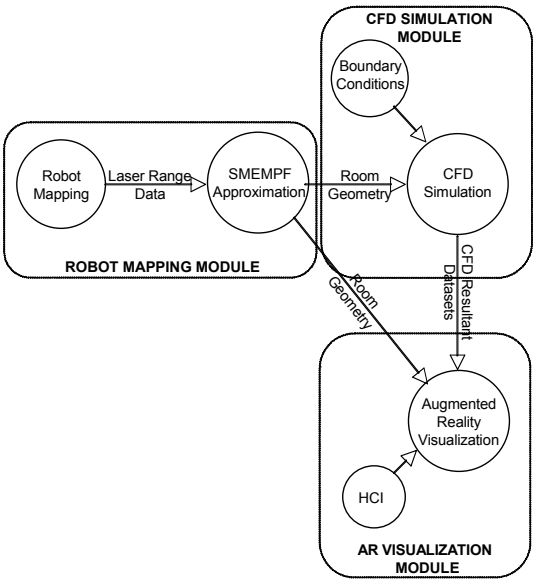


Figure 1: Integrated mobile AR system modules.

The robot mapping module enables the spatial geometry modelling. It employs the SMEMPF method to convert raw 3D datasets to model the spatial geometry. This data is critical for subsequent CFD simulation and AR visualization modules. For CFD simulation, the acquired spatial data is utilized to create a mesh (model setup) that allows model representation for simulation purposes. Any inaccuracies and uncertainties in data will corrupt the credibility of simulation results. For AR visualization, the spatial data is employed to register virtual objects with the real world. Erroneously generated spatial data will amplify registration errors during immersive visualization.

The CFD simulation module facilities prediction of thermal performance data based on laser range information acquired by the robot and boundary conditions. The AR visualization module assists in interactive, immersive visualization of CFD datasets with the aid of see-through Head Mounted Device (HMD), magnetic motion trackers, and gesture-recognition. The advantage of such an integrated AR architecture is the uninterrupted interactive, immersive visualization for multi-room settings.

2.1 Robot Mapping Module

Earlier mobile AR approaches for building performance simulations used hard-coded spatial data [16]. Such implementations render immersive visualizations for multi-room settings difficult. A mobile robot equipped with a laser can aid in gathering precise spatial geometry in real time. The robot incrementally builds a map from laser data that is composed of 3D datasets [17]. A compact environment map composed of few generalized polylines is obtained by converting the raw 3D datasets acquired from a range sensor. This data can be used in CFD simulation and AR visualization modules for interactive, immersive visualization.

To generate steady approximations to robot acquired 3D datasets without assumptions and to avoid local optimal solutions, a *Split and Merge Expectation-Maximization Patch Fitting* planar approximation method was developed. It extends the EM algorithm that provides an iterative solution to compute maximum likelihood estimates given incomplete samples [18].The core of EM procedure is simple least square fitting which is dimension independent, hence plane fitting is the 3D version of line fitting. Three-dimensional EM fits (infinite) planes to the data given. Taking a set of data points in 3D space and an initial set of 2D planes as input, the algorithm alternates two steps, "E-step" (Expectation) and "M-step" (Maximization), until it converges. The algorithm is guaranteed to converge to some *local* optimum.

E-Step	*Given a current set of planes, for each point the probabilities of its correspondences to all planes are estimated based on its distances to planes.*
M-Step	*Given the probabilities computed in the E-step, the new positions of the planes are computed using a regression weighted with these probabilities.*

The SMEMPF method adds two new steps, "split" and "merge," to the EM algorithm. In the first step ("split"), the model components (i.e. the patches) obtained by a previous EM iteration are examined for support of the data points. A higher and homogeneous point density around a patch indicates a presence of a linear structure in the data points. Parts of the components that do not have sufficient support are removed, leading to component splitting and removal. This results in a new set of model components for the next EM iteration. In the second step ("merge"), similar model components are merged together. The "merge" steop prevents generating statistical models that overfit the data, i.e., fit noise in the data. This step requires a similarity measure of patches and is based on the similarity on principles of perceptual grouping used to merge pairs of patches, visually belonging together, to a single patch. The advantage of the SMEMPF method is that the final number of fitted patches is not pre-determined but depends on the objects represented by the data and the extent of noise in the data. In other words, the number of model components is adjusted to achieve the best possible approximation accuracy as a function of noise extent.

The data structure utilized to fit the data is a set of planar rectangles, referred to as *"*patches,*"* which are subsets of the planes computed by the general EM. To be more versatile, each patch is subdivided into a grid of "tiles," figure 2. The number of data points close to the respective tile distinguishes "supported" and "unsupported" tiles. All computations are processed on the set of supported tiles, e.g. the distance of a point to a patch is the distance of the point to the closest supported tile. The grid size (*G)* of a patch defines the length of the edges of its tiles and boundary tiles are resized to fit the patch as necessary. The final result of the modified EM is a set of patches identifying planar macro structures (e.g. walls) that consist of a collection of (coplanar) supported tiles. The tiles identify the shape of these structures in a granularity or resolution determined by the patch's grid size.

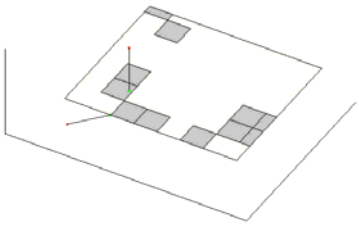


Figure 2: Patch with supported tiles. The outer rectangle is the patch, the dark inside rectangles are supported tiles. All tiles are of same size, with the exception of boundary tiles, being resized to fit the patch.

SMEMPF Planar Approximation Method

The SMEMPF planar approximation method comprises non-reversible Patch Split (PS) and Patch Merge (PM) steps alternating with an EM Patch Fitting (EMPF) algorithm, figure 3. In the PS step, the quality of the EMPF output is evaluated, i.e., how well the EM positioned the new patches. Patches will be split into multiple patches based on the distribution of supported tiles. If a patch contains a large number / area of unsupported tiles, it will be split into multiple coplanar patches to allow a better fit of the supported tiles to the data in the next EMPF step. Hence, PS creates a higher number of patches in order to optimally fit the input data points; the number of patches is *not constant*, as it is in classical EM. The problem of local optimal solutions as appearing in the classical EM due to a wrongly defined number (too low) of fitting patches is overcome, since such a solution will not have a good global support in the data points. In addition to the starting number of patches, the initial position of the patches also does not matter, since the following EMPF will reposition the split patches to better fit the data.

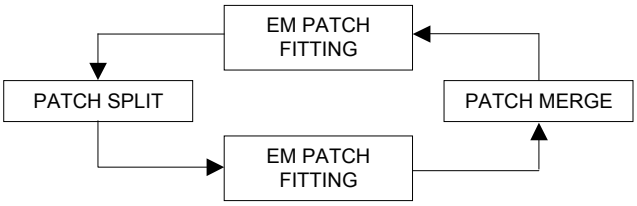


Figure 3: SMEMPF planar approximation method.

In the PM step, the pairs of similar patches are merged to single patches. Due to this process, the number of patches cannot grow to infinity. Hence, the number and position of the new patches introduced by the split is not critical in the modified EM framework. Iterating split and merge in the EM framework is a powerful tool to adjust the number and position of patches to better fit the data points.

After a few iterations, the SMEMPF converges and the PS procedure stops splitting if a certain goodness of fit criterion is met. The stop condition is the stability of distances of data points to the closest patches. At certain instances, in between the iterations, patches are added if a high number of points not supporting any patches are found. This situation can occur if the initial patch setup is placed far away from certain data points. The following subsections discuss in detail the SMEMPF steps – EMPF, PS, and PM, in addition to grid size and experimental results conducted to evaluate SMEMPF's performance.

2.1.1 Expectation-Maximization Patch Fitting

EMPF is modified from the general EM plane fitting to work with patches. It is composed of three steps,

Step 1	***E-step*** *with patches (the EM probabilities are computed based on the point distances to sets of supported tiles).*
Step 2	***M-step*** *with the probabilities computed in the E-step, resulting in infinite planes containing the new patches.*
Step 3	*Trimming planes to patches and determining supported tiles. It includes two sub-steps,* ***Step 3.A**: **Trimming planes to patches** (the planes to patches and tiles by distance projection).* ***Step 3.B**: **Determining of supported tiles** (assignment of supporting data points to planes.*

Step 1: E-step

Let $a_1,..., a_m$ be a set of data points in 3D space, and let $s_1,...,s_n$ be a set of patches. Usually m is significantly larger than n. For each point a_i, the probability p_{ij} that a_i corresponds to patch s_j is computed for $j=1..n$.

Formally, pij = $p(z_i=j)$, where z_i is the hidden variable associated with point a_i whose values range over the patch indices. Analog to EM plane fitting, this probability is computed based on the distance $d(a_i,s_j)$ from point a_i to patch s_j, i.e. the distance to the closest supported tile in patch s_j:

$$p_{ij} \sim exp(- (d(a_i,s_j)^2 / 2S^2) \qquad (1)$$

and normalized so that $\sum_{j=1..n} p_{ij} = 1$ for each i.

The standard deviation (S) in eq. (1) scales the weights p_{ij} with respect to the patch's grid size (G) in a way that points in distance G have a constant weight W (in our system 1/100) before normalization. This guarantees independence from the data points' scale and, since G is decreasing during the iterative EM process, emphasizes the role of local support (i.e. closer) points to determine the planes' positions during the iteration. S is computed by,

$$S = G/sqrt(-2\ log(W)) \qquad (2)$$

(Substituting S in (1) with the right side of (2) yields W for $d(a_i,s_j)=G$).

Therefore, the two differences to the standard EM plane fitting are first the replacement of the distance point to plane with the distance point to closest supported tile in a patch, and second, the use of a distance scaling factor S. After every E-step, a matrix (p_{ij}) is generated, with each row i representing the patch affiliation probabilities for point a_i, also referred to as the "support" of point a_i for the patches s_j. Each column j can be seen as a set of weights representing the influence of each point on the computation of a new patch position in the M-step.

Step 2: M-step

The output of the M-step, which performs an orthogonal regression weighted with (p_{ij}), is a set of (untrimmed) planes $e_1, ... , e_n$ corresponding to the input patches $s_1, ... , s_n$. The normal vector to the plane e_j is the eigenvector to the smallest eigenvalue of the matrix M_j defined as (all sums to be read as $\sum_{i=1..m}$):

$$\begin{vmatrix} \sum p_{ij}(a_{ix}-X)^2 & \sum p_{ij}(a_{ix}-X)\,(a_{iy}-Y) & \sum p_{ij}(a_{ix}-X)\,(a_{iz}-Z) \\ \sum p_{ij}(a_{iy}-Y)\,(a_{ix}-X) & \sum p_{ij}(a_{iy}-Y)^2 & \sum p_{ij}(a_{iy}-Y)\,(a_{iz}-Z) \\ \sum p_{ij}(a_{iz}-Z)\,(a_{ix}-X) & \sum p_{ij}(a_{iz}-Z)\,(a_{iy}-Y) & \sum p_{ij}(a_{iz}-Z)^2 \end{vmatrix}$$

where $a_i=(a_{ix}, a_{iy})$ are the coordinates of the data points, and (X,Y) is their average weighted with p_{ij} for $i= 1...n$. (X,Y) also defines a point on plane e_j , hence the plane e_j is uniquely defined by (X,Y) and M.

Step 3.A: Trimming Planes to Patches

In order to trim the planes to patches with supported tiles, *max.* supporting data points to planes need to be assigned. This assignment is based on the probabilities computed in the E-step. A *support set* $S(s_j)$ for a given plane e_j is defined as a set of points whose probability of supporting plane e_j is the largest (in comparison to other planes),

$$S(e_j)=\{a_i : p_{ij} = max\ (p_{i1}, ..., p_{in})\ \} \qquad (3)$$

A point a_i *supports* a plane e_j if $a_i \in S(e_j)$.

Trimming the planes to patches is a simple step now, using the support set S. For each j, we define the set $S_P(e_j)$ as the set $S(e_j)$ projected (orthogonal) onto the plane e_j. The *trimmed patch* s_j is the minimum bounding rectangle of $S_P(e_j)$, with one edge direction defined by the principal axis of $S_P(e_j)$. A set of patches is generated, $s_1, ... , s_n$ with $s_j \subset e_j$.

The support set for each patch is simply the support set of the corresponding plane, i.e., $S(s_j) = S(e_j)$ for $j =1, ..., n$, a point a_i *supports* a patch s_j if $a_i \in S(s_j)$.

Step 3.B: Determination of Supported Tiles

A patch s_j is decomposed into equal tiles of edge length G (the grid size). For each tile t_k of s_j its support $support(t_k)$ is defined as the number of data points supporting s_j in the cube $C(t_k)$, a cube whose edge length is G, placed around t_k, figure 4.

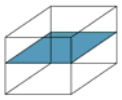


Figure 4: Cube around tile t_k (shaded). All edges have length G.

The union set of points meeting these requirements for all tiles t_k of a patch s_j is referred to as the "reduced support" set of s_j , $S^r(s_j) \subset S(e_j)$, figure 5. In each iteration, a support threshold (T) is computed from the statistics of the $support(t_k)$ values over all tiles of a patch. Tiles tk with $support(t_k) > T$ are marked as supported tiles. If a patch does not contain supported tiles, it is removed.

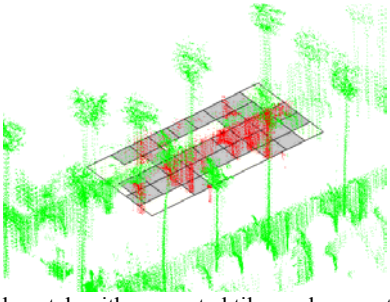
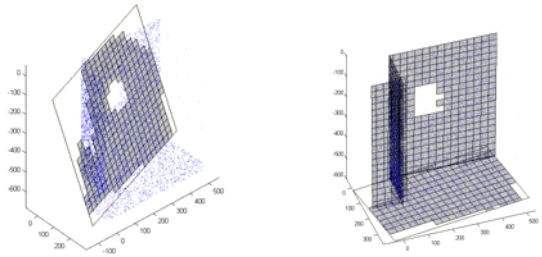


Figure 5: Single patch with supported tiles and support points. The data points (dark shade) show the reduced support set S^r of the patch.

The parameters G and T are computed dynamically each time in the trimming step. G is computed as $3\ std(d)$, where $std(d)$ is the standard deviation of point patch distances computed for all data points to the patches they support. T is computed as $mean(c) - 2\ std(c)$, where c is the number of points in the reduced support set S^r. It should be noted that computing T needs to be done with the current G and not the prior. This can be achieved by recounting points in $C(t_k)$ before computing T.

2.1.2 Patch Split

A classical case of the EM local optimum problem is illustrated in figure 6.



A) B)

Figure 6: Fitting data without and with splitting. A) fitting the dataset with one patch (result of classical EM with initial model of 1 patch). B) after splitting into 4 patches using SMEMPF.

The problem in figure 6A is that only one patch is used, while four patches are needed. 6B shows the result using SMEMPF, automatically gaining the required number of patches. Splitting is processed along axes in the patch having insufficient support of data points, figure 7. First the points $a_i \in S^r(s_j)$ of the reduced support set of patch s_i are projected onto the patch, yielding a point set in a 2D coordinate system defined by s_i. In this 2D system, the points a_i are projected onto the X-axis under different rotations (0,45,90,135 degree) to gain density information along the respective directions. On the X-axis, they are quantized into bins of size G, the grid size. The bin containing the minimum amount of points min_p defines position and direction of a split axis. If min_p falls below a threshold $min_p < 2/3(\#S^r(s_j))$, ($\#$ = number of points), a split is caused. The split divides $S^r(s_j)$ into two sets of points $S^{r+}(s_j)$ and $S^{r-}(s_j)$, left and right of the split axis. Two new patches s_j^+ and s_j^- are created in the plane e_j, the plane containing s_j. Using $S^{r+}(s_j)$ and $S^{r-}(s_j)$, they are trimmed and their supporting tiles are determined. s_j^+ and s_j^- then replace the original patch s_j in the model. The split procedure is recursive: s_j^+ and s_j^- again are processed the same way, until each resulting patch has a sufficiently homogeneous distribution of supporting points and is not split further.

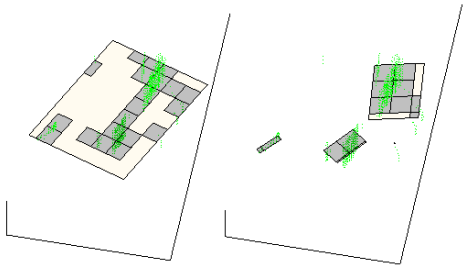


Figure 7: Split. Left the original patch, right the split patches. The three new patches are slightly replaced compared to their origin, due to new patch directions determined by the principal axis of projected support points

The number of points in $S^r(s_j)$ depends on the grid size G, which therefore indirectly steers the splitting process. A smaller G enables reaction to more local density differences, and the split resolution is higher (due to smaller bin size). The resulting new patches are

coplanar with the original patch s_j, split determines a new *number* of patches, a number that leads to a better fit to the data due to the *replacement* of the newly created patches in the follow up EMPF step.

2.1.3 Patch Merge

The iterated EMPF followed by PS only, without merge, could possibly grow the number of patches with each iteration to a potentially large number. Therefore, merging "similar" patches is necessary, figure 8. PM is responsible for the accuracy of the statistical model; without it the model may end up fitting the noise. If a given patch is properly split, then EMPF will reposition the resulting patches to better fit the data points in a way that will turn them away from each other. If a patch is unnecessarily split, the patches remain very similar after an EMPF iteration, where similar means that they will be nearly collinear and close to each other. This suggests that merging should combine two perceptually similar patches to a single one, and leave unchanged perceptually dissimilar patches.

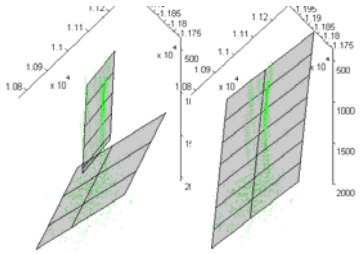


Figure 8: Merging of two patches (left) to a single one (right). The merged patch results from fitting a single plane to the union of supporting points of the two original patches, followed by trimming.

Merging is derived from 2D line merging algorithms based on principles of perceptual grouping [19]. Intuitively, the underlying similarity measure takes into account the closeness, coplanarity, and angle between normals of two patches. The similarity, therefore, is based on the model, not on the dataset to be fit. Merging of a single pair s_a , s_b of patches is processed in two steps,

Step 1	*Similarity determination.*
Step 2	*If sufficiently similar, merge by creating a single patch using least square fitting and trimming.*

Step 1: Similarity Determination

For similarity determination, the patches have to be sufficiently close. This is defined by checking for overlap of bounding cubes of each patch, expanded by G (grid size) in each direction. If this condition is met, then the patches are tested if they are also sufficiently coplanar and parallel. To determine this, the points $a_i \in S^r(s_a)$ supporting s_a are projected onto the plane e_b containing s_b and vice versa ($b_j \in S^r(s_b)$ are projected onto e_a). The mean distance of the points to their projections is computed for both sets, resulting in two values d_a, d_b. If $min(d_a, d_b) < G$ the patches will be merged.

Step 2: Merge

A new plane is fit to the set union $S^r(s_a) \cup S^r(s_b)$ with a classical regression and trimmed to a patch. Merging is done iteratively until all possible pairs of patches are sufficiently dissimilar.

2.1.4 Determination and Influence of the Grid Size Parameter (G)

A central parameter in the SMEMPF method is the grid size G. Apart from its visible manifestation as the edge length of each tile of a patch, it controls different instances in the process such as, the tiling size and the distinction between supported and unsupported tiles. Indirectly, by determining tile support, it steers the split process; and it determines candidates for merging.

G is newly determined in each iteration of the EMPF process, based on the distribution of the distances of data points to patches/tiles they support. It is computed as *3std(d)*, where *std(d)* is the standard deviation of point patch distances computed for all data points to the patches they support. G can be seen as quality of approximation of the patches to the data; a lower value shows a better approximation. This measure of quality is only a valid measure if the number of patches is fixed. Increasing the number of patches, e.g. to a value such that each patch is exactly supported by three data points, yields $G=0$, but simply measures overfit to noise instead of data. Experimental results show that the interplay between split and merge leads to a decreasing value of G in each iteration step in the beginning of the SMEMPF process (when starting with a high value) until it balances itself. The decrease is due to two reasons,

- *The re-computation of G is made using the reduced support set. Since this is a subset of the closest points in the support set, the standard deviation is smaller than the standard deviation of the (unreduced) support set.*
- *But primarily, split leads to a better fit.*

In contrast, the merge process increases the standard deviation due to a less optimal fit. Hence, as long as, intuitively, there's a stronger rate of split than merge, as it naturally is in the first iteration steps if the process is started with a low number of patches, G decreases until it balances itself to an appropriate value due to the increasing force of merge during the iteration.

2.1.5 Experimental Results

The final result of the SMEMPF method consists of coplanar point sets, orthogonal projection to project the reduced support sets $S^r(s_j)$ onto their supported patches s_j was used. The results, therefore, can be seen as a segmentation of the original point set into sets of replaced (projected) data points, related by their support to the same patch. Non-assigned data points are seen as outliers and are dropped. To determine the performance of the SMEMPF method, an experiment was conducted.

The experiment generates 25 sets $N_{1..25}$ of ~7000 points each by randomly sampling a 3D model of four walls with a different amount of Gaussian relocation (replacement noise), figure 9. The standard deviation of point distances to the ground truth walls is 1-25 for N_1 to N_{25} respective, i.e. $std(N_i)=i$. Therefore, the representation of the walls by the point clouds is more blurred in higher indexed data sets, figures 9 C,D,E for examples of N_2, N_{15}, N_{23}. The length of the short edges of the walls forming the small corner (front left in figure 7A) is 150 units; therefore, the structure of this corner gets lost in the point cloud representation with high standard deviation (15-25 units, point sets N_{15} to N_{25}). For each set $N_{i=1..25}$ the SMEMPF is processed, initialized with a set of three patches, figure 9B. Similar to the evaluation used in [10], the result is analyzed by summing the angular distances of all patch normals to their corresponding ground truth, giving the quality of approximation. Figure 9F shows a graph of the resulting differences for the 25 sets. It can be seen that the approximation is constantly good even up to a high noise level (std=15, see figure 9 D,F). The value of a difference of ~4 degrees

shows an average difference of 1 degree for each patch. The jump in values for N_{22} to N_{23} results from wrong fitting: the high amount of noise destroyed the corner feature, figure 9 E, F.

As the robot performs range scanning to obtain an environment map using the SMEMPF planar approximation method, the acquired datasets are employed in the subsequent two modules for simulation and visualization purposes.

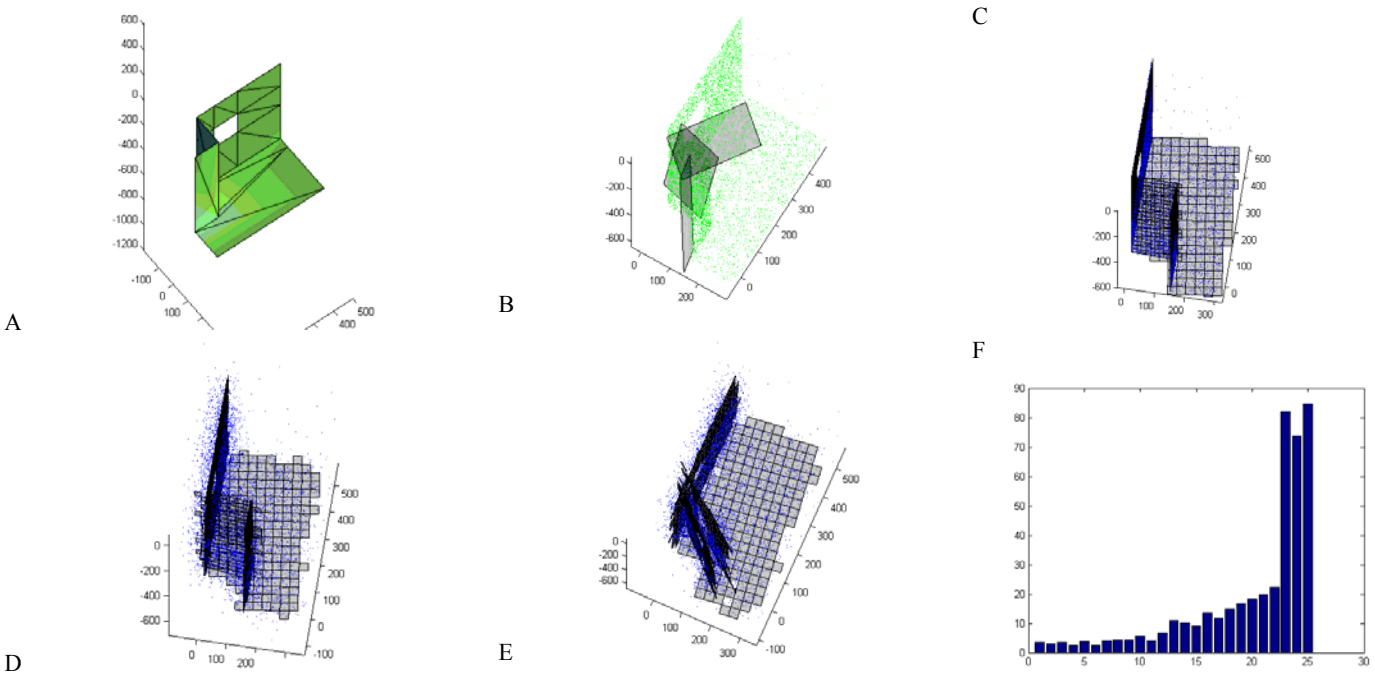
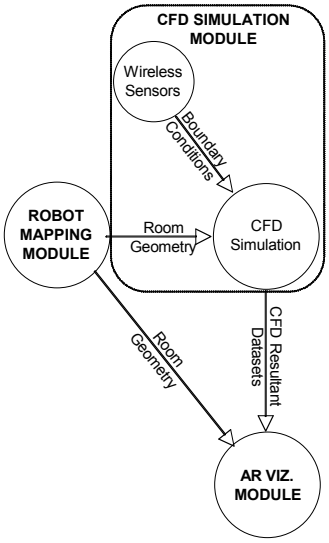


Figure 9: Fitting to a ground truth dataset. A) the original model (4 walls) B) set (no noise) of simulated scan points created from A) and initial 3 patches. The figures C-E show the result of the SMEMPF algorithm (each after 6 iterations), using the initial patch configuration shown in B) but with random replacement of scan points. C) replacement with standard deviation of 2, D) std = 15, E) std= 23. The length of the short walls being misrepresented in E) is 150; the representation fails due to the high amount of noise, and the original structure is not visible in the point set. F) Sum of angular errors of patch normals. Y-axis: angular error, x-axis: standard deviation of replaced data points. The values at x=2, 15, 23 correspond to C), D) E)

2.2 CFD Simulation Module

CFD simulation module is used to predict 3D indoor thermal behavior for use by AR visualization. CFD simulation for indoor environments involves two major steps, modeling the room geometry and setting initial / boundary conditions, figure 10. The room geometry is modeled using robot mapping. The SMEMPF method employed allows for identifying patches with precision. These patches are then utilized to develop a computational model that represents the room geometry. The boundary conditions for CFD simulation are applied based on real-time wireless sensors that track temperature and velocity changes within the room. These changes occur as users interact with the actual space or as the space condition changes due to environmental fluctuations within the room. As the indoor thermal environment changes, the wireless sensors update the boundary conditions of the CFD simulation module.



The space is, then, discretized into uniform grids. The simulation progresses based on the convergence criteria. CFD simulation utilizes Fluent 6.02 and Gambit 2.0.6 [20]. Once the simulation converges, the resultant datasets are forwarded to the AR visualization module for immersive visualization.

Figure 10: CFD simulation module: room modeling and boundary conditions.

2.3 AR Visualization Module

The AR visualization module tracks the user's movement in real time and poses graphical representations of CFD simulation datasets on the HMD for the user to visualize and interact with. This module consists of two sub-components, the multimodal HCI and AR pipeline. The multimodal HCI enables efficient data manipulation by users while still being immersed in the visualization of CFD datasets, in actual space. It consists of a library of speech and gesture recognition tasks that aid in data manipulation. IBM Viavoice [21] was employed for speech recognition. For gesture recognition, CyberGlove [22] was used. It captured global hand motion using trackers attached to the glove and local fingers' motion as a set of joint angles. Using custom-prepared functions, the hand posture data was transformed into commands that allowed data manipulation.

The AR sub-component consists of tracking of user's head motion, generating new perspective graphical representations of resultant CFD datasets based on tracker data, and posing this information to the HMD in real time, figure 11. Flock of Birds [23] magnetic trackers were used to track user movement. To alleviate the issues inherent to AR such as registration errors due to metals present in the room and latency, Gaussian and Kalman filters were employed [24].

Figure 11: AR visualization module.

3.0 CONCLUSIONS

The paper discussed a framework for integrated AR architecture for indoor thermal performance data visualization using mobile robot for environment mapping. It presented a *Split and Merge Expectation-Maximization Patch Fitting* planar approximation method to achieve robust visualization. The SMEMPF, used as a mapping procedure in robotics as a combination of Expectation Maximization patch fitting with alternating patch splitting and merging, was proven to be a powerful tool to gain a patch representation of maps formerly consisting of independent laser range scanner reflection points. The newly introduced merging step balances the number of patches, created by splitting, in a visually natural way and therefore allows for the number of starting patches for the EM step to be highly imprecise.

Although the present study demonstrates the potential for such integration, issues related to automatic transitions between the spatial data generated by the robots and the simulation engine need to be further explored. Methods of translations between the spatial data, simulation, and augmented visualization need to be developed.

4.0 REFERENCES

[1] Bryson S [1993] Virtual Wind Tunnel: A high-performance virtual reality application. In Proceedings of IEEE Annual Virtual Reality International Symposium, 20-25.

[2] Impelluso T [1996] Physically Based Virtual Reality in a Distributed Environment, Computer Graphics, 30(4): 60-61.

[3] Wasfy TM and AK Noor [2001] Visualization of CFD Results in Immersive Virtual Environments, Advances in Engineering Software, 32: 717-730.

[4] Malkawi AM and RS Srinivasan [2005] A New Paradigm for Human-Building Interaction: The Use of CFD and Augmented Reality, Automation in Construction Journal 14 (1): 71-84.

[5] Malkawi AM [2004] Immersive Building Simulation, Chapter in Advanced Building Simulation, AM Malkawi and G Augenbroe, (eds), Spon Press, UK.

[6] Thrun S, D Fox and W Burgard [1998] Probabilistic mapping of an environment by a mobile robot. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA), 54

[7] Hahnel D, D Schulz and W Burgard [2002] Map Building with Mobile Robots in Populated Environments. In Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systems (IROS).

[8] Wang CC and C Thorpe [2002] Simultaneous Localization and Mapping with Detection and Tracking of Moving Objects. In Proceedings of the IEEE Int. Conference on Robotics & Automation (ICRA).

[9] Thrun S, C Martin, Y Liu, D Hähnel, R Emery-Montemerlo, D Chakrabarti and W Burgard [2003] A Real-time Expectation Maximization Algorithm for Acquiring Multi-planar Maps of Indoor Environments with Mobile Robots. IEEE Transactions on Robotics and Automation.

[10] Triebel R, W Burgard and F Dellaert [2005] Using Hierarchical EM to Extract Planes from 3D Range Scans. In Proceedings of the IEEE International Conference on Robotics and Automation (ICRA).

[11] Weingaren J and R Seigwart [2005] EKF-based 3D SLAM for Structured Environment Reconstruction. In Proceedings of IROS, Edmonton, Canada.

[12] Denis F, Wolf, A Howard and GS Sukhatme [2005] Towards Geometric 3D Mapping of Outdoor Environments Using Mobile Robots. In Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 1258-1263.

[13] Früh C and A Zakhor [2000] An Automated Method for Large-Scale, Ground-Based City Model Acquisition. In International Journal of Computer Vision, 60(1): 5 – 24

[14] Sack D and W Burgard [2004] A Comparison of Methods for Line Extraction from Range Data. In Proceedings of the 5th IFAC Symposium on Intelligent Autonomous Vehicles (IAV).

[15] Veeck M and W Burgard [2004] Learning Polyline Maps from Range Scan Data Acquired with Mobile Robots. In Proceedings of the IEEE/RSJ Int. Conference on Intelligent Robots and Systesm (IROS).

[16] Malkawi AM, R Srinivasan and JV Veer [2005] Interfacing with Building Data: Toward an Integrated Mobile AR Environment. In Proceedings of the 9th IBPSA Building Simulation Conference, Montreal, Canada.

[17] Lakaemper R, LJ Latecki, S Xinyu [2004] Geometric Robot Mapping. In Proceedings of the 12th Intl. Conference of Discrete Geometry for Computer Imagery (DGCI).

[18] Dempster, N Laird and D Rubin [1997] Maximum Likelihood from Incomplete Data via the EM Algorithm. Journal of the Royal Statistical Society, Series B, vol. 39(1).

[19] Wertheimer M [1923] Maximum Likelihood from Incomplete Data via the EM Algorithm. Psycologische Forschung, vol. 4: 301–350.

[20] Fluent Inc [2003] Website accessed on June 10, 2003. URL: http://www.fluent.com

[21] IBM ViaVoice [2003] Website accessed on March 26, 2003. URL: http://www-306.ibm.com/software/voice/viavoice/

[22] Virtual Technologies Inc. [2003] Website accessed on August 17, 2003. URL: http://www.immersion.com/

[23] Ascension Technologies Inc [2003] Website accessed on June 20, 2003. URL: http://www.ascention-tech.com/

[24] Malkawi AM and R Srinivasan [2005] Building Performance Visualization using Augmented Reality. In Proceedings of the 14th Intl. Conference on Computer Graphics (GRAPHICON).

About the Authors

Rolf Lakaemper is an Assistant Professor at the Department of Computer and Information Sciences, Temple University, Philadelphia. His contact email is lakaemper@knight.cis.temple.edu

Ali Malkawi is an Associate Professor at the Department of Architecture, University of Pennsylvania, Philadelphia. His contact email is malkawi@design.upenn.edu

Ravi Srinivasan is a PhD candidate at the Department of Architecture, University of Pennsylvania, Philadelphia. His contact email is sravi@design.upenn.edu

Longin Jan Latecki is an Associate Professor at the Department of Computer and Information Sciences, Temple University, Philadelphia. His contact email is latecki@temple.edu