Visualization of Indoor Thermal Environment on Mobile Devices based on Augmented Reality and Computational Fluid Dynamics

Jia-Rui Lin\textsuperscript{a,b,1,*}, Jun Cao\textsuperscript{c,d,1}, Jian-Ping Zhang\textsuperscript{a}, Christoph van Treeck\textsuperscript{c}, Jérôme Frisch\textsuperscript{c}

\textsuperscript{a}Department of Civil Engineering, Tsinghua University, China
\textsuperscript{b}National Engineering Laboratory for Green and Safe Construction Technology in Urban Rail Transit, Tsinghua University, China
\textsuperscript{c}Institute of Energy Efficiency and Sustainable Building (E3D), RWTH Aachen University, Germany
\textsuperscript{d}Research Center of Green Building and New Energy, Tongji University, China

Abstract

Augmented reality (AR) based visualization of computational fluid dynamics (CFD) simulation on mobile devices is important for the understanding and discussion of indoor thermal environments in the design process. However, utilizing AR-based mobile device for indoor thermal environment understanding still encounters problems due to limited computational power and lack of efficient interaction methods. To improve the performance of AR-based CFD visualization on a mobile device and provide the users with intuitive interaction with indoor environment, an integrated approach based on client-server framework is established. Within the approach, a new mobile friendly data format (cfd4a) with low computational complexity is proposed for CFD simulation results representation. Server-side data pre-processing method is also introduced to reduce the computational power and time needed on the client side. Furthermore, interactive section view selection and time-step animation methods are proposed for intuitive interaction with the AR environment. Then, a prototype system is developed with Unity3D engine and Tango Tablet. Demonstration in

*Corresponding author

Email address: jiarui_lin@foxmail.com, lin611@tsinghua.edu.cn (Jia-Rui Lin)

\textsuperscript{1}These two authors contribute equally to this study.
some typical scenarios shows that the proposed method and prototype system provide an intuitive and fluent AR-based environment for interactive indoor thermal environment visualization. Comparing to the widely used vtk format, a data compression ratio of 63.4% and a loading time saving ratio of 89.3% are achieved in the performance test with the proposed method. The proposed approach also shows good flexibility and extensibility in supporting different AR devices through exposing standard web services and utilizing the widely used Unity3D engine. Source code of the developed prototype and relevant testing data are also shared through github, enabling other researchers to compare our work with theirs. It is also suggested that stability of AR-based mobile devices, new interaction methods and integration with cloud computing still need further investigation and improvement.

**Keywords:** Augmented Reality (AR), Computational Fluid Dynamics (CFD), Indoor Thermal Environment, Mobile Device, Performance Improvement, Client-Server Framework

1. **Introduction**

Due to its high impact on energy saving and occupants’ health, design and simulation of indoor thermal environment are attracting more and more attention in the architecture, engineering and construction (AEC) area [1, 2]. On one hand, building energy saving has been required due to growing interest in global environmental issues [3], on the other hand, accidents caused by indoor heat stroke and heat shock also calls for renovation of buildings and living spaces for a more comfort indoor environment [2, 4]. Thus, it is important to adopt numerical simulation methods like computational fluid dynamics (CFD) to predict and assess indoor thermal environment [3, 5] both in design and renovation phase.

However, as stated by Fukuda et al. [2], indoor thermal design is not performed in an efficient manner by the owners, architects, facility managers, etc. One big reason is that the simulation results generated by CFD software appli-
cations are composed of massive numerical data and hard to understand by non-professionals. Generally, results of a CFD simulation is visualized as two dimensional drawings, and are not easy to comprehend for people that are not familiar with them. Moreover, simulation results are always separated with real-world objects, that is, the relationship between the simulation results and physical indoor environment is hard to be established and understood by the owners, facility managers, and other non-professional stakeholders. As a result, stakeholders’ participation in the design process of indoor thermal environment and collaboration efficiency are highly influenced.

To solve the above-mentioned problems, virtual reality (VR) environment based on a game engine is proposed to visualize the simulation results. Within a VR environment, indoor thermal environment can be visualized as arrows and color maps, which can help the client to understand the indoor airflow more intuitively. Meanwhile, distributed meeting is also possible with this kind of VR application, and face-to-face discussions can be reduced. However, 3D objects in a VR environment are disconnected from the real-world objects, which makes it difficult for the stakeholders to get a direct understanding of the relationship and interaction between the virtual data and the real world. Therefore, augmented reality (AR) is introduced more recently to combine virtual simulation results with real-world environment. Using an AR environment, correct and effective visualization of CFD simulation results can be achieved, thereby reducing the misinterpretation in spatial and logical aspects. The near real-time interaction with these virtual and real objects also enables a designer or client to check multiple parameters simultaneously and analyze and discuss the problem effectively and intuitively. In this manner, design decision-making can be performed with high accuracy based on CFD simulation and VR/AR environment instead of designers’ experience. Recently, AR-based environment is explored for design collaboration. New slicing and interaction methods are also introduced to visualize CFD data in an AR environment.

Nevertheless, AR-based CFD visualization for indoor thermal environment
is still not satisfactory for practice. There is a trend for AR to transit from
desktop to mobile devices, whose portability and easy propagation provide the
engineers or users with convenient access to relevant information in situ[10].
However, only a few projects have investigated the potential of AR technology
on a mobile device for CFD visualization[15, 10]. A visualization module for
CFD simulation is implemented based on Unity3D engine[16], and 3DS Max
is used to pre-process the 3D data and the simulation results are converted to
TXT or CSV files. Similar exploration for AR-based visualization of finite el-
ment analysis on mobile devices is also conducted[17]. As concluded in[10],
most of the current research for AR-based visualization of numerical simulation
like CFD on mobile devices mainly centered on the feasibility and applicability
of AR technology on mobile devices, and limitations including computational
power, rendering defects still remain to be resolved[18]. Since CFD results are
huge numerical data sets, efficient conversion and processing of these massive
data sets to a visualization friendly format is essential[2] on mobile devices,
calculating the virtual contents from CFD results on a mobile device with lim-
ited computational power will take too much time and may annoy the users or
designers. Furthermore, most of the studies only take AR as a visualization
tool, effective and intuitive interaction methods to interact with the simulation
results are not fully explored[10, 14].

To improve data visualization performance and explore new interaction meth-
dos for AR-based CFD visualization on mobile devices, after guiding the reader
through the literature review for CFD, AR as well as their application for indoor
thermal environment design, the paper proposes an integrated approach based
on client-server framework. A mobile friendly data format and server-side CFD
data pre-processing are introduced to reduce computational time and transfer
time used for mobile devices. Meanwhile, interactive section view selection and
time-step animation methods are also proposed to help the user interact with
the AR environment intuitively. Finally, with demonstration in a use case exam-
ple and collected user feedback, the potential benefit, future applications and
improvements of this technology are concluded.
2. Literature Review

2.1. Computational Fluid Dynamics (CFD)

CFD adopts numerical analysis and corresponding data structures to analyze and solve problems related to fluid flows. It is widely used to simulate indoor thermal environment or particle flow patterns. There exist many computational fluid dynamics frameworks and programs ranging from problem solving environments using open source software frameworks such as OpenFOAM [19] to widely applied commercial software packages such as ANSYS Fluent [20]. In the course of this work, a software package called ‘MPFluid’ developed by Frisch [21] is applied. Generally, fluid flows are modeled based on the Navier–Stokes equations. Complete derivations of the equations are investigated by Frisch [21] as well as previous researchers [22, 23, 24]. The CFD simulation usually requires a high computational power, and can be carried out on a massive parallel computation system. The above-mentioned MPFluid has proven very good scalability results up to 140,000 processes on two German Supercomputers SuperMUC and JuQueen [25]. After the simulation, the data post-processing and visualization is usually done by using ParaView [26] based on the visualization toolkit (VTK) library [27]. The underlying file format is ‘vtk’, which is widely used in scientific visualization as the geometry can be modeled using a lot of primitives and it can be enriched with resulting scalars, vector fields, or tensors. With simulation results exported as vtk files from MPFluid, further data post-processing can be done based on the simulated velocity fields using the VTK library. For example, streamlines, cuts (slices) or contour plots can be obtained and saved as vtk file for future visualization in an AR environment.

2.2. Augmented Reality (AR)

AR is a technology that [11]: 1) combines real and virtual content, 2) is interactive in real time, 3) is registered in 3D. Thus, an AR system is considered as next generation, reality-based interface supplementing the real world with virtual objects, which appear to coexist in the same space as the real world [28].
Furthermore, the system also generates interactive graphics responding to user input in real time and tracks its position to maintain relative locations of the virtual image and the real world [11].

Nowadays, AR technology has matured and proven to be an innovative and effective solution to help solve some of the critical problems in simulation, collaboration, planning, construction, and maintenance [29, 30] with the following benefits:

1. Provisioning of an intuitive way to interact directly with information.
2. Improved understanding of design decisions and better perception of the real world.

An essential technology to exploit the benefits of the AR system is the position and motion tracking of devices and users [18]. Sensor-based, vision-based, and hybrid tracking techniques [31] are adopted for real-time position and motion tracking. The basic idea is to compute and to estimate the position based on sensor data, visual features, and 3D models.

Meanwhile, hardware and toolkit development for AR is also an important factor for AR-based applications. Head-mounted Display (HMD) devices have been applied widely for a direct perception of the combined AR scene [29]. Hand-held devices and projectors are further explored to decrease uncomfortable feelings like headache, dizziness and nausea [29]. At the same time, to help in application creation, 3D rendering engines such as Unity3D or Unreal Engine have integrated necessary packages for AR development in their software development kits (SDKs).

To ensure real-time interaction between virtual information and real world, AR devices need capabilities in useful data filtering, fast graphics display, and accurate and reliable motion tracking [18]. Even if many benefits and considerable advances are made, some limitations with the technology remain to be overcome. Nowadays, position drifting, occlusion detection and 3D registration are still key problems of any AR system [18]. The problems will be more critical on a mobile device, more efforts are needed to resolve problems regarding com-
putational power, visualization defects and motion tracking that obstructing the adoption of AR.

Recently, to give mobile devices a human-scale understanding of space and motion, Google proposed a prototype device with highly customized hardware and software called Project Tango [32]. Project Tango is a smart device equipped with a variety of cameras and vision sensors, enabling real-time depth perception and object tracking as well as instantaneous 3D mapping [33] that really enables and improves AR and its various applications. With the release of the Tango device, C/C++/Java APIs and Unity3D engine SDK integration are provided [34] to facilitate application development.

2.3. Augmented Visualization of CFD Simulation

Due to its benefits for better interaction and understanding of the virtual design and the real world, AR is taken as a new paradigm for human-building interaction [8] since 2005. However, above-mentioned problems regarding computational power, visualization defects and motion tracking still obstruct the adoption of AR in the CFD simulation domain. Until recently, with development both in hardware and software, some explorations were conducted to apply AR for CFD applications. Yabuki et al. [7] adopted AR to visualize environmental simulation results for collaboration. AR environment as well as building information modeling (BIM) were also integrated for design feedback in thermal environment design [3, 9]. Later, related work was extended for renovation of buildings and living spaces [2]. Similar to AR, mixed reality was as proposed for visualization of CFD simulation [12]. Meanwhile, interaction methods for numerical simulation [14] and slicing methods based on stylus [13] were introduced for more interactive AR environment. On mobile devices, Unity engine is used to develop a visualization module for CFD simulation [16], and 3DS Max is adopted to pre-process the 3D data and the simulation results are converted to TXT or CSV files. Similarly, exploration for AR-based visualization of finite element analysis on mobile devices is also conducted [17]. Among different AR developing tools, Unity3D stands out due to its benefits including free for
non-commercial use, platform independent, high performance and easy to learn \cite{35}. For example, Unity3D was used to create an interactive and immersive 3D environment to visualize and communicate CFD simulation results between users of varying experience \cite{16}. Meanwhile, Unity3D was adopted to visualize CFD results in a VR environment for indoor thermal design feedback \cite{9}. With AR-based visualization of thermal environment, engineers, designers and the owners can examine the simulated results and the environment interactively, and communicate the design more efficiently \cite{10}.

Nowadays, AR is undergoing a transition from desktop to mobile devices, with which the designers, users or engineers can get relevant information conveniently in situ \cite{10, 36}. Since the computational power, storage space, capacity of the network are limited, AR with mobile devices requires new methods that can access the data visualize it more efficiently \cite{36}. CFD simulation usually generates massive numerical data sets, and data transformation is needed to visualized them in an AR environment \cite{2}. This transformation will pose additional computational load to the mobile device and may take too much time, which would annoy the users and affect the communication efficiency during design process of indoor thermal environment.

2.4. Summary

As a common technology for understanding indoor thermal environment, CFD is widely used and usually generates quite a lot of data for further processing and visualization. By enabling interaction between the virtual and real world, AR brings a new paradigm to comprehend indoor thermal environment. However, due to limited computational power and lack of efficient interaction methods, AR-based CFD visualization on a mobile device are not fully explored and need further investigation. This is also a motivation factor of this work for addressing these shortcomings.
3. Methodology

Considering the limited computational power, complex data processing tasks are not acceptable on mobile devices. Therefore, large scale CFD simulation data should be pre-processed on work stations or servers and then shared with mobile devices. Based on this idea, an integrated approach based client-server framework is proposed (Figure 1).

Within the proposed approach, a simplified mobile-friendly data format for representation of CFD simulation results called cfd4a, is established (Middle of Figure 1). When designing the cfd4a format, data visualization requirements like geometric mesh size, direct data section access, etc. are taken into account to minimize the data computational power used on mobile devices.

Meanwhile, data pre-processing method is introduced to convert the widely used vtk format to the proposed cfd4a format. There are three main steps for data preprocessing: vtk loading that loads the data into memory, data merging that merges possible duplicate geometric data in vtk files, and data slicing that divides the data into small pieces to fulfill limitation of mobile devices. With the consideration of computational power, data pre-processing tasks are implemented on the server-side. Except for data pre-processing, a web service based on representational state transfer (REST) protocol is adopted. Though later mentioned in the data pre-processing section, data compression is encapsulated in REST services. That is, when a hypertext transfer protocol (HTTP) request is received, the service module can route different requests to different service functions, and compress the data if required before sending it to the client. Left part of Figure 1 illustrates the data pre-processing and hosting method based on the file management features of Windows Server.

While, the client-side is built based on AR SDK and Unity3D SDK for Android, and also possible for other mobile platforms (Right part of Figure 1). When requested data is received from server-side, data decompression is conducted to inflate the data and the loaded into the AR environment. Data caching is also considered to cache received data in local storage, so that we do not need
to ask for the same data multiple times. The most important part proposed for the client-side resides on mobile devices is the visualization and interaction method. Since the CFD simulation data can be aligned to the coordinate system, an axis-aligned section view selection method by simply touching on the screen is introduced. In addition, data switching method between different time-steps is also proposed to animate dynamic changes of CFD simulated indoor thermal environment.

Figure 1: Integrated approach for AR-based CFD visualization on mobile devices

4. Implementation

4.1. CFD data pre-processing for mobile devices

Since the computational power is limited on mobile devices, data pre-processing should be done on the server-side. In this subsection, a mobile device friendly format, cfd4a, is proposed to fulfill the requirements for CFD visualization on mobile devices. Then, corresponding data pre-processing methods is introduced to convert the widely used vtk format to cfd4a. The following contents describes the disadvantages of current CFD data formats, the advantages of the newly derived data structure, and the data conversion processing.
4.1.1. Requirements for CFD visualization on mobile devices

As mentioned above, AR-based data visualization on a mobile device faces many difficulties. One of the most difficult challenges is the limited memory and calculation capacity of the mobile device. Even though performance of central processing unit (CPU) and graphics processing unit (GPU) on a mobile device has improved considerably, its capacity for 3D visualization is far from the potential of a desktop computer or even a cluster system. Thus, some HMD for AR-based visualization either need a desktop computer assisting in data visualization or provide a visualization with low resolution only. Therefore, the available limited memory and computation capacity of mobile devices should be used only on the most critical part for AR-based visualization of CFD data.

On the other hand, to control memory usage and improve 3D visualization efficiency, Unity3D uses Mesh objects to keep 3D data inside its own application. As stated in the Unity3D scripting API [37], a Mesh object has several important attributes: vertices, normals, uv, colors, indices and meshTopology. The first four attributes are a collection of vectors containing position, normal direction, texture coordinate and color of a vertex. The last two attributes provide information on how to construct geometry primitives (such as points, lines, triangles) based on different vertices, and meshTopology specifies the type of geometry primitive and the topology to form the given geometry primitive. Using a Mesh object enables the removal of duplicate vertices and, thus, reducing the overall memory consumption. In addition to further reduce the geometry data density, a Mesh object of Unity3D takes a 2-byte short data type to describe each item in the collection of indices. This implementation limits the amount of the vertices to a maximum of $2^{16} - 1$ (65535) [38].

4.1.2. cfd4a: simplified data format for AR-based CFD visualization on mobile devices

In the CFD area, vtk files are widely used for CFD data representation and sharing. However, since vtk format is a common file format designed to offer a consistent data representation scheme for a variety of dataset types, different
geometry topologies and data attributes are involved, and each component of these data has a complex structure. Thus, the complex nature of vtk format requires additional procedure and computational power that are quite limited on a mobile device, and leads to the following deficiencies when applying native vtk as underlying data format for AR-based CFD visualization on a mobile device:

1. Different time steps in different vtk files leads to geometry data duplication;
2. Complex structures require more computation power to transform a vtk file based geometry and attributes to Unity3D;
3. Lack of ability to go directly to a specific data attribute, thus, more time is necessary for file scanning to find a specific data attribute;
4. Data can be further compressed with gzip or other algorithms to reduce storage consumption and time for data transfer.

In order to improve CFD data visualization efficiency on a mobile device, a mobile-friendly data format with nearly no data duplication, less pre-processing work and direct data section accessing ability needs to be provided. Therefore, a new binary file format called cfd4a (short for CFD data format for Android) is proposed. As shown in Figure 2, the overall structure of the format can be divided into four sections: Header, Attribute Definition, Submeshes, and Attributes. Components and structures of these sections are further described as follows:

1. **Header**: The header is designed to store basic information of a CFD simulation result. These information include version, name, bounding box of the CFD data, amount of time steps, amount of submeshes, and amount of attributes contained in the CFD data. In addition, the starting points to read submeshes are also provided in the Header section. Hence, the reader can skip the Attribute Definition section and go directly to the Submeshes section, if necessary.
2. **Attributes Definition:** This section of the format is used to describe name, type and starting position of each attribute in the CFD simulation results. The number of the attributes can be found in the Header section. Thus, switching between different attributes of a CFD simulation only requires to parse the Header and Attribute Definition section and then go directly to the corresponding data position of the desired attribute.

3. **Submeshes:** The Submesh section is a collection of submeshes. The definition of each submesh in this section is similar to the structure of a Mesh object in Unity3D, thus, transforming a submesh to a Mesh object in Unity3D is straightforward and fast. In addition, given that different time steps of a CFD simulation may involve different submeshes, an additional component called time step is introduced to describe the starting time step from which on the submesh should be visualized.

4. **Attributes:** Similar to the Submeshes section, the Attributes section consists of a series of attributes. Parts of each attribute include submesh, time step count, item count, component count, as well as type and raw data array. The first four parts describe the link of an attribute to a submesh, how many time steps and attribute items are contained in this attribute, the component amount of each attribute item, and the data type of each component.

In the newly defined cfd4a format, complex representations for geometric topology is stored as a submesh structure in cfd4a format. As the submesh structure is very similar to the structure of a Mesh object in Unity3D, the time for data conversion can be reduced considerably. Since multiple vtk files for different simulation time steps are merged into one single cfd4a file, redundant geometric information are removed, and the memory consumption can be streamlined. By introducing a time step property in the Submesh structure, relationships between geometric data and time step information are maintained. The dataset for point attributes and cell attributes in the vtk format is also simplified as an attribute structure in the cfd4a format. Furthermore, proper-
ties such as submesh start and attribute start are introduced as fast data access cursors. This enabled the skipping of unused data sections and reduces the time for file loading considerably.

4.1.3. Conversion from vtk to cfd4a

To decrease data pre-processing work at the client side, raw CFD simulation results saved in vtk format need to be converted into cfd4a and hosted at the sever side. The conversion process consists of six procedures: vtk file loading, geometry dataset conversion, attribute dataset conversion, multiple time step combination, splitting data into small pieces, and finally cfd4a file saving (see Figure 3).

1. vtk loading: load raw vtk data into memory and prepare for data conversion.
2. Geometry dataset conversion: this step involves the conversion of different geometric topologies into a uniform submesh structure.
3. Attribute dataset conversion: convert attribute dataset using different data types and structures into a simplified structure in cfd4a format.

When processing cell attributes in vtk format, data interpolation is re-
quired to convert cell attributes into point attributes.

4. Multiple time step combination: combine multiple time steps stored in different vtk files into one. First, identify redundant geometric data by comparing the geometric topology of different time steps. Then, remove redundant geometric data and combine corresponding attribute data with necessary time step information attached.

5. Data splitting: since index properties of a Mesh object in Unity3D are implemented as a collection of 2-byte short, the maximum amount of vertices in a Mesh object can not exceed 65535. Therefore, a procedure is required that splits large CFD data models into several smaller ones.

6. cfd4a saving: save the converted CFD data into cfd4a format.

![Figure 3: Process for converting vtk files into cfd4a files](image)

4.2. RESTful API for CFD data hosting at the server side

To manage CFD results of different simulation configurations or different projects and make it possible to serve multi-platform clients, standard data services based on widely used RESTful API is established. In this way, the client can access the required data at any position that has internet access.

In a RESTful API service, http verbs such as GET, POST, PUT, DELETE are adopted for data manipulation operations, all the data hosted at server side are taken as resources, and identified by their URLs. Usually, GET is used for querying the data, POST and PUT are used for data creation and updating, while DELETE is applied for data removal. When requests made
to a resource’s URL, the service will elicit related data or files and send it to the client side as a response. Considering the requirements for CFD result visualization and corresponding data hosting service, a RESTful API is designed with the following functions:

1. Building Information Model (BIM) hosting if the CFD simulation is based on BIM.
2. Point cloud model service for hosting scanned data from real world.
3. View slice service that provides different section views of a CFD simulation.
4. Tube data service for hosting of streamline data representing air flow patterns.

The related API definitions are listed in Table 1.

<table>
<thead>
<tr>
<th>Route</th>
<th>Functions</th>
<th>Supported operations</th>
</tr>
</thead>
<tbody>
<tr>
<td>/bim/</td>
<td>list available BIM models</td>
<td>GET</td>
</tr>
<tr>
<td>/bim/{id}</td>
<td>manage BIM models</td>
<td>GET, POST, DELETE</td>
</tr>
<tr>
<td>/scan</td>
<td>list available point cloud models</td>
<td>GET</td>
</tr>
<tr>
<td>/scan/{id}</td>
<td>manage point cloud models</td>
<td>GET, POST, DELETE</td>
</tr>
<tr>
<td>/slice</td>
<td>list available slice views</td>
<td>GET</td>
</tr>
<tr>
<td>/slice/{id}</td>
<td>manage slice views</td>
<td>GET, POST, DELETE</td>
</tr>
<tr>
<td>/tube</td>
<td>list available streamline views</td>
<td>GET</td>
</tr>
<tr>
<td>/tube/{id}</td>
<td>manage streamline views</td>
<td>GET, POST, DELETE</td>
</tr>
</tbody>
</table>

Table 1: Available RESTful API operations

4.3. Augmented reality-based CFD visualization on mobile clients

Key feature of an AR environment is the overlaying of virtual data onto the real world and interacting with virtual and real world simultaneously in real-time. Hence, a transformation for matching virtual data on real world data needs to be determined based on different technologies such as 3D registration
or computer vision. Meanwhile, to visualize CFD simulation results at different location and show the dynamic change of indoor thermal environment, both section view selected and animation are necessary. However, the method for interaction with AR-based CFD simulation on mobile devices is not deeply explored. Therefore, an automatic axis-aligned section view selection method is implemented, and the user can choose a section by simply touching on the screen. Then, time-step animation is also introduced to animate the simulation.

When implementing the proposed methods, it is assumed that the point of origin of the coordinate system is identical to the center of axis-aligned bounding box of the virtual data set.

### 4.3.1. Positioning virtual data in real world

As illustrated in Figure 4, the data positioning has to find a function that maps the bounding box of the virtual coordinate system \( X - Y - Z \) from the CFD simulation to real-world coordinate system \( X_r - Y_r - Z_r \) tracked through the Tango Device. In other words, data positioning can be achieved by identifying a matrix to transform \( X - Y - Z \) to a local coordinate system \( X' - Y' - Z' \) of \( X_r - Y_r - Z_r \). The calculation can be split into the calculation of translation, scaling and rotation. Generally, to determine such a transformation matrix, at least four non-coplanar points are necessary. To simplify the calculation process, four typical points are chosen in a specific order from the corners of the bounding box as depicted on the right-hand side of Figure 4. The resulting three point pairs define the forward, up and right direction of the local coordinate system.

In this pattern, the center \( P_c \) of the bounding box \( X_r - Y_r - Z_r \) is determined by calculating the mid-point of \( P_1 \) and \( P_4 \) (Equation 1), describing the center of the virtual CFD data set. The result is stored in point \( P_c \). The scale of the virtual data is determined from the distance between \( P_1 \) and \( P_4 \) and the diagonal distance \( (L_{diag}) \) of the bounding box of the virtual CFD data (Equation 2). Thus, the orientation of the virtual data in real-world can be obtained by calculating the forward, right and up direction (Equation 3).

\[
P_c = \frac{(P_1 + P_4)}{2}
\]  

(1)
\[ S = Distance(P_4, P_1)/L_{\text{diag}} \quad (2) \]

\[
\begin{align*}
V_{\text{forward}} &= \text{Normalize}(P_2 - P_1) \\
V_{\text{up}} &= \text{Normalize}(P_3 - P_2) \\
V_{\text{right}} &= \text{Normalize}(P_4 - P_3)
\end{align*}
\] (3)

Figure 4: Virtual CFD data positioning

4.3.2. Automatic section selection for CFD flows

Usually, CFD simulation results are volumetric datasets and it is difficult to visualize the inner and outer surface at the same time. Creating different section views or slices at critical positions is common strategy for volume data visualization. In an AR environment with better interaction between a human and the scene mixed with virtual and real world, an ideal way for volume data visualization is to simultaneously create section views or slices with the changing of the view point and facing direction of the device. However, this kind of interaction needs much more computational and visualization power, which is currently not affordable for a mobile device such as the Tango tablet. Therefore, a method for automatic slice selection based on the touch point is proposed to maintain better interaction capacities. The basic idea behind this method is to first create section views along different directions with a proper interval, and
hosting of all the CFD section views at the server side. The system can then figure out which section view is the best one related to the position the user touched and request a transfer.

Considering the normal use case for CFD data visualization in an AR environment, the CFD simulation results and corresponding BIM or point cloud model are usually axis aligned or can be easily transformed into an axis fashion. The most common scenario is navigating in the AR environment to view and communicate with the CFD simulation results and the design. Therefore, when creating section views of the CFD data, only the directions parallel to axis $X$, axis $Y$, and axis $Z$ are necessary. Taking the interval along axis $X$ and $Y$ as 0.5 meter and the interval along axis $Z$ as one meter is adequate for AR-based CFD visualization in this use case. To help section view identification for data hosting, all section views are numbered in a sequential order with a prefix to reflect its normal direction. For instance, section view $x2$ identifies the second slice with its normal parallel to the axis $X$. Note that sequential numbers of section views start from 1 at the minimum side of the bounding box to the maximum side along an axis (a top-down view is shown in Figure 5). With this consideration, a variety of section views are created and hosted at the server side in the above-mentioned cfd4a format through a RESTful API service.

To find the best-fitting section view based on the user’s touch, the axis-aligned bounding box of the CFD data is utilized as well (Figure 4). Once the user touches a point $P_t$ in the mixed scene, a ray from the view point (or the position of camera in Unity3D) $P_v$ to the touching point is created. Then, the intersection information between the ray and the bounding box is determined. The position of the intersected point $P_i$, the intersected plane $Plane_{ymax}$, and the normal direction $n$ of $Plane_{ymax}$ are all included in the intersection information. With these information, the system can calculate the normal direction of the section view based on $n$ and up direction of the camera, in Figure 5 a section view with its normal direction parallel to axis $X$ then needs to be loaded. By calculating the relative distance between the intersection point $P_i$ and $P_{xmin,ymax}$ (or $P_{xmax,ymax}$), the required section view can be determined.
based on the pre-defined interval distance between two section views. Finally, based on the normal direction and the sequential number of the section views, a http request is built, which is sent to the server to access the data of the specific section view. Thus, the section view $xm$ depicted in Figure 5 is downloaded from the server and displayed in the device.

![Figure 5: Find proper section view for CFD data](image)

4.3.3. CFD attributes visualization

Attributes visualization in an AR environment is similar to a common 3D visualization. Scalar attributes or any component of vector attributes are mapped to a range of colors, and vector attributes can be visualized as arrows to reflect the direction and magnitude. Another widely used method are streamlines or tubes created from streamlines to represent the airflow in a space. In this paper, colors are adopted to represent the scalar attributes (such as temperature or air speed magnitude) and streamlines represent the airflow direction.
4.3.4. Simulation time step animation

Multiple time steps of a CFD simulation are embedded into one single cfd4a file. Hence, only one http request is necessary to download a specific section view or streamline set from the server. Once a cfd4a file is downloaded, all geometric data and data related to selected attributes are loaded into memory to support fluent switching between different time steps. When loading the data, the color range corresponding to the attribute is created based on the minimum and maximum value of all data related to the specific attributes across different time steps. After loading, the first time step is activated, which means all geometries related to the first time step is set visible and their color is set according to the attribute data related to the first time step. Switching to the next time step involves deactivating geometries not related to the current time step, activating geometries related to the current time step, and finally refreshing the colors of all the active geometries according to attributes data of the current time step.

5. Results

To prove the concept of the proposed methodology, a server-client prototype system for AR-based CFD visualization is developed based on C# language, Unity3D engine and Tango SDK.

The server side is implemented with NancyFx [39], a lightweight web service library built on the .net framework. For testing the developed prototype, the server side is deployed on Windows Server 2012. BIM, point cloud and CFD simulation results are hosted at the server side. The client side is developed on the basis of Unity3D and Tango C# SDK. Then the client is packed as an Android application and deployed on the google Tango tablet. Finally, the Tango tablet is connected to a wireless network provided by the host.

The BIM, point cloud, and CFD simulation results used for the demonstration are created and gathered based on an office room at RWTH Aachen University. The point cloud model of the office is created using the Tango tablet;
the BIM model of the office was created in Autodesk Revit 2016, while the CFD simulation is based on the scanned point cloud model. To simplify the test and validation process, all kinds of models related to the test case are imported into ParaView, then transformed and aligned to a unified coordinate system, hence, involving currently some manual work.

5.1. Use case definition for CFD simulation and visualization

As briefly mentioned before, the applied use case consists of a typical office room in one of the RWTH Aachen University buildings, which can be seen in Figure 6. It can be noticed that a lot of additional furniture is present and, thus, this room can be considered as a typical, realistic setup.

![Figure 6: Pictures of the testing room](image)

(a) Direction A  (b) Direction B

In order to be able to simulate the room in MPFluid, a scan of the room geometry is performed using the Google Tango device depth sensors and capturing capabilities. The geometry is stored and transferred from the device to a computer using the polygon file format PLY. This raw geometry information is used in the MPFluid internal voxelization approach transferring a surface geometry into a volume geometry. This conversion process is described in detail in [21]. The CFD model of the room shown in Figure 6 consists of approximately 1.4 million cells with a cell size of 3.03 × 3.70 × 2.85 cm in x, y, z direction, respectively. Once the volume geometry is created, boundary conditions are applied to the model describing the room situation as exactly as possible with
respect to input flow velocities at windows or temperature boundary conditions. Then, the simulation was performed on a local cluster using 16 processes only, as a low resolution CFD simulation suffices for the current applications.

5.2. AR-based visualization

5.2.1. Basic workflow

With the developed TangoCFD client, four steps are involved for overlaying virtual data on real-world scene (see Figure 7). The first step is putting markers by touch that represents the four points used for transformation calculation described in detail in Section 4.3.2 (see Figure 7a). After the transformation mapping based on the created points, virtual data like the BIM, the point cloud and the streamlines are loaded and positioned at proper locations (see Figure 7b). Then, if a section view of the CFD simulation should be loaded, the user can touch the screen (see Figure 7c) and the client calculates the position of the required slice. Finally, the relative position of the slice to the bounding box are sent to the server to download corresponding section view data and visualize it (see Figure 7d).

5.2.2. Overlay of CFD data on BIM/point cloud

According to the workflow for CFD data visualization described above, the developed prototype was first tested to visualize CFD simulation results with a point cloud model and subsequently with BIM. When conducting this test, a small corner of the real world is used to create the required markers for virtual data positioning, so that the user can get an overall view of the CFD data and the BIM or point cloud model.

In this scenario, users such as engineers can get a third-person view of the overall virtual model and corresponding CFD results, which can be used in discussion of the simulation. As shown in Figure 8 after loading the point cloud model in the AR environment, users can touch the screen to load section views with air speed attributes along $x$ or $y$ direction, and discuss the distribution of the air speed. At the same time, it also possible for the users to load tube-based
streamlines start from the window (one of the CFD boundary conditions with inflow air velocity), and see the evolution of the airflow patterns by switching between different time steps. For example, time steps 1 to 4 in Figure 8 show clearly the development of the airflow starting from the window going to the inside of the room before leaving through the open door in the rear of the room.

Similar results are illustrated in Figure 9 as well. With the BIM model, furniture and layout of the office are clearly provided. The furniture in the room has a significant influence on the airflow, as displayed in Figure 9b. When switching between different view positions, like zoom in and out of the scene, details of the simulation results and BIM model can be visualized intuitively. When conducting this test case, Tango shows good ability in motion tracking most of the time, which means the relative position between the virtual and real world is well handled.
5.2.3. Overlay of CFD data on the real world

To further explore the ability and capacity of an AR environment for CFD visualization, another scenario that overlays the simulation results on the real scene is investigated. The user can see the airflow illustrated by streamlines enter the room through the window based on the developed AR system (displayed in Figure 10). The airflow direction and speed changes in accordance with the simulation time are also clearly revealed. By overlaying a slice on the real scene, the influence on the air flow of indoor obstacles can also be depicted (Figure 11b). However, when loading multiple slice views and tube-based streamline, the visual defects shown in Figure 11 can be observed. For example, part of the slice view should be occluded by the desk and chair in the room, which is currently not correctly handled by Tango tablet and the corresponding API. Another problem is that it is hard for a user to get an overall view of the simulation.
5.3. User Feedback

To further validate the proposed system, 8 users are invited to test the prototype and provide their comments. As listed in Table 2, 4 graduate students, 2 designers, and 2 building owners are invited, half of them are familiar with CFD simulation, and 75% of them have already tried HMD-based VR environment. Before testing the proposed prototype, a detailed explanation on how to use it is conducted by the authors. Then, the above-mentioned two typical scenarios, namely, scaled-down third-person view and full-scale view, are tested one after another by all the invited users. Finally, an interview is conducted for each person. Their comments and feedback on the intuitiveness, immersiveness and flexibility as well as the performance and interaction methods of the proposed approach are collected.

Regarding the intuitiveness and immersiveness of the prototype based on
Project Tango, 100% positive feedback is collected. It is agreed that understanding CFD simulation of indoor environment is easy and straightforward in such an AR environment. The owners and 2 students also emphasize that seeing the interaction and relationship between the virtual simulation data and the real-world environment is valuable for non-professionals. As virtual simulation and real-world environment coexist in AR environment, the designers and the owners feel that they can discuss and communicate more effectively and explanation of the simulation can be reduced. Except for the benefits, one designer also noticed that the section view created from CFD simulation is always floating on the real-world, even though it should be occluded by the desk in the room. This visual defect will lead to misunderstanding of the relative location of the virtual data and the real world sometimes.

Similar feedback on the interaction methods is received from the users.
users found that loading a slice by a simple touch on the screen is easy and intuitive, no extra time is needed to learn it. Moreover, most of the users said that chaining different time steps of CFD simulation as a short animation in an AR environment makes it easier and intuitive to understand how the air flows and how the thermal environment changes. Users tried HMD-based VR environment also commented that an AR environment based on mobile devices can free them from the wires and they can move in the indoor space more freely. Some users also mentioned that loading cfd4a files is much faster than loading the raw vtk files.

5.4. Performance Test

Meanwhile, data loading time of raw vtk format and the proposed cfd4a format on the client side are measured and compared, and the proposed cfd4a format shows a huge advantage. As shown in Table 3 comparing to native
<table>
<thead>
<tr>
<th>Category</th>
<th>Familiar with CFD</th>
<th>Tried VR Before</th>
</tr>
</thead>
<tbody>
<tr>
<td>4 Students</td>
<td>2 Yes, 2 No</td>
<td>4 Yes</td>
</tr>
<tr>
<td>2 Owners</td>
<td>2 No</td>
<td>1 Yes, 1 No</td>
</tr>
<tr>
<td>2 Designers</td>
<td>2 Yes</td>
<td>1 Yes, 1 No</td>
</tr>
</tbody>
</table>

Table 2: Information of invited users

vtk data format, cfd4a takes only 10%-40% of the time for CFD data loading (X, Y slice and streamline in this test case), which is reasonable for mobile device. Another benefit of cfd4a files is that usually it take less storage space than raw vtk files (Table 2). Further analysis shows that CFD data with same geometric data in multiple time steps will get a high compression ratio and loading efficiency. For example, the X slice has a compression ratio of 66.9% and a loading time saving ratio of 89.3%. These improvements are achieved by avoiding multiple file reading, content duplication and whole file scanning. Therefore, BIM model and point cloud data with no attribute data have a low loading time saving, and CFD streamline with different geometric data of multiple time steps have a relative low compression ratio, hence, a data compression algorithm such as gzip is introduced. With data compression at the server side, and data decompression and caching abilities at the client side, network traffic is carefully controlled to support fluent CFD data accessing for augmented visualization. As mentioned above, cfd4a is a simplified format that lost the ability to represent various geometric topologies and multiple attribute data types. This is a trade-off among data loading time, complex CFD data representation as well as computational power and space, users and researchers should consider the importance of each aspect and choose a proper solution.

5.5. Discussion

Demonstration and user feedback of the proposed prototype based on Tango tablet illustrate that a visualization of CFD simulation results based on AR provides a comprehensive view by combining the virtual and real world. It is
Table 3: Loading time (ms) comparison between vtk and cfd4a

<table>
<thead>
<tr>
<th>Data</th>
<th>Content type</th>
<th>vtk</th>
<th>cfd4a</th>
<th>Ratio</th>
<th>cfd4a (zipped)</th>
<th>Ratio (zipped)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIM model</td>
<td>geometric data</td>
<td>77</td>
<td>16</td>
<td>79.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Point cloud</td>
<td>geometric data</td>
<td>355</td>
<td>237</td>
<td>33.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$y$ slice</td>
<td>same geometric data and different attribute data of multiple time steps</td>
<td>1,689</td>
<td>180</td>
<td>89.3%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x$ slice</td>
<td>same geometric data and different attribute data of multiple time steps</td>
<td>1,301</td>
<td>141</td>
<td>89.2%</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Streamline</td>
<td>different geometric and attribute data of multiple time steps</td>
<td>4,716</td>
<td>1,847</td>
<td>60.8%</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4: File size (kB) comparison between vtk and cfd4a

<table>
<thead>
<tr>
<th>Data</th>
<th>vtk</th>
<th>cfd4a</th>
<th>cfd4a (zipped)</th>
<th>Ratio</th>
<th>Ratio (zipped)</th>
</tr>
</thead>
<tbody>
<tr>
<td>BIM model</td>
<td>382</td>
<td>245</td>
<td>77</td>
<td>35.9%</td>
<td>79.9%</td>
</tr>
<tr>
<td>Point cloud</td>
<td>4,296</td>
<td>3,576</td>
<td>1,572</td>
<td>16.8%</td>
<td>63.4%</td>
</tr>
<tr>
<td>$y$ slice</td>
<td>15,612</td>
<td>5,165</td>
<td>1,643</td>
<td>66.9%</td>
<td>89.5%</td>
</tr>
<tr>
<td>$x$ slice</td>
<td>11,700</td>
<td>3,856</td>
<td>1,259</td>
<td>67.0%</td>
<td>89.2%</td>
</tr>
<tr>
<td>Streamline</td>
<td>74,439</td>
<td>60,021</td>
<td>14,354</td>
<td>19.4%</td>
<td>80.7%</td>
</tr>
</tbody>
</table>

also easier to understand the influence of real world environment on the air flow patterns itself. At the same time, the positioning of virtual models by a few point markers and loading slice view by touches on the screen are easy and intuitive and such interactions will improve the usability of the AR environment.

It is also worth noting that the Tango tablet provides a stable motion tracking feature and effectively reduces the occurrence of position drifting.

At the same time, the proposed cfd4a format and corresponding data conversion method can reduce the loading time dramatically. Generally, if results of a CFD simulation can be exported or converted into vtk format, then it is pos-
sible to pre-process and persist the data in cfd4a format on the server side, and visualize them in a AR environment. Automated slice selection is also possible under this situation.

Additionally, this research also makes a step forward in scalability and multi-platform support. Most of the current studies support only one platform instead of multi-platforms, lack of scalability restricted the application range of these systems\[10\]. Since the adopted visualization engine, Unity3D, is widely used in different mobile devices and has various AR extensions, it is also possible to reuse the server side completely and most of the client side code to create AR-based CFD visualization tool for other mobile devices.

Furthermore, when developing and testing the prototype system using the Tango tablet, the device is heating up quickly, and sometimes crashes. Obviously, the Tango device is not stable enough yet, but it provides the user with a much better motion tracking ability to build robust AR environments. Due to the limitations on the memory and computational power, data size and data pre-processing time of the CFD results need to be carefully controlled, in order to get a reasonable 3D rendering performance. In addition, other visualization strategies or methods can be further investigated in future work.

Inevitably, limitations for AR-based CFD visualization which is not specific to Tango tablet still exist in some scenarios. As highlighted in the demonstration, overlaying CFD data on the real-world test case means scaling the virtual model to the same size as the real world. This prevents the user to get an overview experience of the whole simulation results, which can be necessary from time to time. Moreover, most of the AR devices just overlay virtual model onto the real world, but cannot visualize their relative location and occlusion relationship correctly sometimes. Scenarios like part of the virtual model occluded by real world can not be correctly handled with these AR devices, which leads to a weird result. Good news is that Tango support occluding point cloud by real-world components \[34\]. Though complex surface-based model is not correctly handled in Tango now, we believe that the occlusion problem will be resolved in the near future.
6. Conclusion and Future Work

To improve usability of mobile device for AR-based visualization of indoor thermal environment, an integrated methodology was proposed. By introducing a mobile-friendly data format and transferring heavy data pre-processing work to the server-side, requirement for computational power at the mobile client is carefully controlled. Meanwhile, with proposed data positioning methods of the framework, easy interaction for overlay virtual data on the real world is achieved. Therefore, with the developed prototype based on an AR-ready device and a data server, better visualization results for different kinds of data resulting from CFD analyses are achieved. It is possible to interact with the reality more intuitively than before on a mobile device, thus enabling a better understanding and insight of the CFD simulation and the design. Since the proposed method adopts the Unity3D engine and exposes standard RESTful API, it is also possible and flexible to incorporate other AR-based devices such as Meta 2, HoloLens. Moreover, to make it easier for the readers to compare our implementation with theirs, the source code and relevant testing data is uploaded to github (https://github.com/LinJiarui/ARvis-CFD).

Possible improvements of this research may include automatic data positioning based on 3D registration, other new visualization or interaction strategies for CFD, and utilization of more features provided by the API of AR devices such as Google Tango. With 3D registration algorithms like iterative closest point\cite{40}, transformation between point cloud used for CFD simulation and point cloud captured during AR-based visualization, therefore enabling automatic positioning of CFD results. Meanwhile, evaluations of AR devices with joy-pad or gesture support are also very valuable for finding intuitive and easy-to-use interaction methods for virtual object manipulating\cite{41}. Furthermore, as AR devices evolving very quickly, new abilities like area learning of Google Tango continue to appear, whose benefits for CFD simulation understanding should always be explored. Finally, considering the emerging trends of artificial intelligence (AI), cloud computing and their applications, integrating AR-based CFD visualiza-
tion with intelligent data retrieval[42] and cloud-based collaboration[43] could also be a good idea, which will enrich the AR environment with more engineering information.

7. Acknowledgment

This work was funded by the National Key R&D Program of China (Grant No. 2017YFC0704200) as well as by the Excellence Initiative of the German federal and state governments, the Beijing Natural Science Foundation (No. 8194067), the Young Elite Scientists Sponsorship Program by China Association for Science and Technology (No. QNRC2016001) and the Tsinghua University-Glodon Joint Research Centre for Building Information Model (RCBIM).

References


URL http://papers.cumincad.org/cgi-bin/works/Show?id=ecaade2015_83


URL http://dl.acm.org/citation.cfm?id=2874916.2874929


URL https://openfoam.org

URL http://www.ansys.com/Products/Fluids/ANSYS-Fluent


URL https://www.paraview.org


URL https://hal.archives-ouvertes.fr/hal-01530500


URL https://docs.unity3d.com/ScriptReference/Mesh.html

URL http://www.stefangordon.com/pyritecli-unity-vertex-limit/

URL http://nancyfx.org/


URL https://doi.org/10.1007/978-3-642-32645-5_34

