Lagrangian Coherent Structures for Design Analysis of Revolving Doors

Benjamin Schindler, *Member, IEEE CS*, Raphael Fuchs, Stefan Barp, Jürgen Waser, Armin Pobitzer, Robert Carnecky, Krešimir Matković, *Member, IEEE CS*, Ronald Peikert, *Member, IEEE*

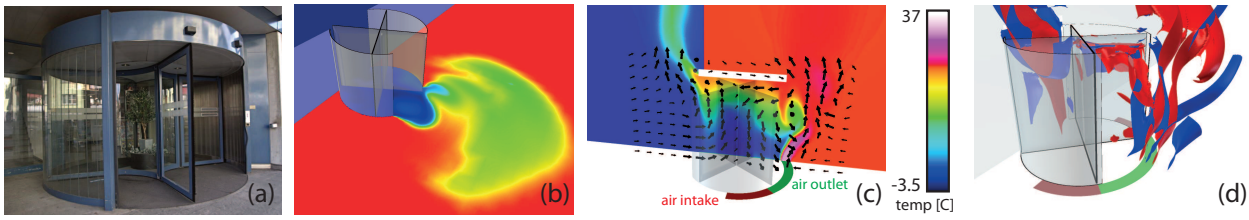

Fig. 1. A revolving door (a), as is typically used in building entrances. The simulation of the air flow (b) near such a door reveals how large swaths of cold air enter the building, incurring energy losses. Image (c) shows how the addition of an air curtain blowing warm air blocks cold air, preventing it from cooling down the interior. This, however, complicates the flow structure greatly so that neither temperature plots nor contours, path- or streamlines really help in understanding the flow behavior. The solution to this problem are LCS (d), giving structural information about this complicated time-dependent flow.

Abstract— Room air flow and air exchange are important aspects for the design of energy-efficient buildings. As a result, simulations are increasingly used prior to construction to achieve an energy-efficient design. We present a visual analysis of air flow generated at building entrances, which uses a combination of revolving doors and air curtains. The resulting flow pattern is challenging because of two interacting flow patterns: On the one hand, the revolving door acts as a pump, on the other hand, the air curtain creates a layer of uniformly moving warm air between the interior of the building and the revolving door. Lagrangian coherent structures (LCS), which by definition are flow barriers, are the method of choice for visualizing the separation and recirculation behavior of warm and cold air flow. The extraction of LCS is based on the finite-time Lyapunov exponent (FTLE) and makes use of a ridge definition which is consistent with the concept of weak LCS. Both FTLE computation and ridge extraction are done in a robust and efficient way by making use of the fast Fourier transform for computing scale-space derivatives.

Index Terms— Visualization in physical sciences and engineering, topology-based techniques, vector field data.

1 Introduction

Energy efficiency in building technology has become a major topic when dealing with today's climate challenges. Several types of simulations provide the possibility of analyzing and improving the energy efficiency of a building prior to construction, including computational fluid dynamics (CFD) for simulating the indoor air flow. In many countries, standards exist for so-called passive houses or, more generally, low-energy-consumption buildings. Some authorities subsidize private buildings if they meet such a standard, while for public buildings, a standard can be declared mandatory. One important factor for energy efficiency is building climate control and, in particular, efforts to minimize energy loss through air exchange.

In building climate control, preventing air draft is one of the foremost issues when trying to reduce heat leakage. It occurs when there are two openings to the outside within a building. The resulting draft not only causes warm air to leave the building but also cold air to enter the building. Specially shielded windows and doors are commonly used and prevent most of the leakage when closed. Air exchange systems are then put in place, which supply the interior with fresh air constantly and at much lower energy losses than if windows were opened to get fresh air into the interior.

In large public buildings such as shopping malls, this task becomes more difficult as people constantly move in and out. Revolving doors are frequently put in place in such cases as they prevent direct air exchange with the outside. They also have the advantage of not letting in snow, rain or dust.

Another common technique to reduce draft is air curtains placed inside the doors, usually blowing warm air from the side or from the top. Air curtains can significantly reduce the flow incoming cold air as well as the outgoing warm air. This is also important for the health of people working in proximity of these doors.

Both of these approaches, taken alone, have certain disadvantages. Air curtains, used with sliding doors, do not prevent the direct contact of the warm air inside with the cold air outside. In large buildings where these doors open frequently, enormous energy losses still occur. Also, the placement of the curtains is difficult. A small change in the environment can result in pressure gradients bending the air curtain. That in turn will stop the air barrier from working, and cold air can enter the building. With revolving doors alone, the problem is that cold air is mixed with warm air inside the doorway and released into the building periodically. Even though draft is prevented, heating is required to compensate for the heat loss due to mixing.

In combination, the two technologies of revolving doors and air curtains have been shown in CFD simulations to achieve significantly better energy saving results [2]. The use of air curtains is especially interesting in buildings adhering to a low-energy-consumption standard, even though it seems to contradict intuition.

In this paper, we tackle the problem of analyzing the simulated air flow near a revolving door equipped with an air curtain. The air flow to be analyzed and visualized has a complicated time-dependent struc-

- *B. Schindler, R. Fuchs, R. Carnecky and R. Peikert are with ETH Zurich, E-mail: {bschindler,raphael,crobi,peikert}@inf.ethz.ch*
- *S. Barp is with Air Flow Consulting AG Zurich, E-mail: barp@afc.ch*
- *J. Waser and K. Matković are with VRVis Vienna, E-mail: {jwaser,matkovic}@vrvis.at*
- *A. Pobitzer is with University of Bergen, E-mail: armin.pobitzer@uib.no*

Manuscript received 31 March 2012; accepted 1 August 2012; posted online 14 October 2012; mailed on 5 October 2012.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org.

ture, because it is the result of both convection and the pump-like effect of the revolving door. A visualization of the temperature alone, by means of slices or isosurfaces, shows the temperature distribution over time. This is one of the end results the engineer is most interested in, besides energy loss and velocity of the draft inside the building. However, the cause of this temperature distribution is not explained by its visualization. For this, a visualization is needed in order to reveal structures in the velocity field. Vector field topology is a candidate, but it is known to not give much insight in flows which are sufficiently unsteady [33]. Even though the air flow caused by a revolving door in combination with an air curtain is nearly steady in some areas, it is highly unsteady around the air curtain, which is the region of primary interest. As a result, vector field topology is not an ideal candidate for the visualization of such simulations and Lagrangian coherent structures (LCS) should be used instead. LCS have been introduced as the material surfaces with maximal stretching and are regarded as a possible replacement for separatrices from vector field topology. LCS rely on finite-time Lyapunov exponents (FTLE) which, briefly summarized, are a measure for flow stretching in a finite time period. We will show how LCS add to a better understanding of flow patterns which are relevant as they affect the efficiency of the air curtain. For a brief overview of the application case, see Fig. 1.

Besides the analysis of the air flow, we propose a number of algorithmic improvements to the LCS extraction and rendering. It was found before [25] that ridge extraction and, in particular, the extraction of ridge surfaces from 3D data requires smoothed derivatives. By using a convolution with derivatives of a Gaussian kernel, consistent derivatives of the required orders are obtained. Instead of computing the convolution explicitly, the fast Fourier transform (FFT) is used, and derivatives are computed in the frequency domain. This leads to a significant overall speed up, while producing high quality gradients, leading to smooth ridge surfaces. Ridge surfaces have to be restricted to regions where the FTLE value is reasonably high. Therefore, the triangle mesh is trimmed at a certain FTLE threshold. Also, our ridge definitions assume that the FTLE gradient is roughly tangential to the surface, which however cannot be guaranteed. To improve the output quality even further, the triangle mesh is trimmed a second time to cut away parts where the FTLE gradient deviates from the surface tangent. Also, since ridges are generally not orientable, a post-processing operation ensures orientability of the extracted meshes. We have implemented the aforementioned techniques in the Visdom visualization framework [38].

The remainder of this paper is structured as follows: In the next section, we cover related work in the areas of building climate control and LCS extraction. Then we introduce our application case and describe our algorithm for extracting the LCS in detail. In the results section, we provide an in-depth analysis of the flow behavior observed including a small heat leak found using the LCS-based visualization. Together with our domain experts, we propose two design optimizations, which greatly minimize the size of the leak. Also, the performance of the entire LCS extraction process is discussed. We conclude by covering some ideas for improving LCS-based visualizations.

2 Related Work

Visualization in Building Climate Control: While indoor air flow has always been an important application area of CFD, visualization of the results was typically done with standard flow visualization techniques [31, 18]. An evaluation of various flow visualization techniques for the purpose of indoor climate design has been done by den Hartog [7] in 2003. Besides classical techniques, he included vector field topology and vortex glyphs [27] and came to the conclusion that these two techniques are more relevant than most standard techniques, but not as relevant as pathlines and isosurfaces. Publications on visualization of indoor climate data often favor scalar field visualization techniques such as isosurfaces, in particular of temperature and velocity magnitude [5, 2]. Staubli et al. [35] used direct volume rendering for visualizing simulated smoke propagation. Due to the underlying model, the extinction coefficient depends linearly on the particle concentration, thus the integrated opacity transfer function is well defined and can be precomputed.

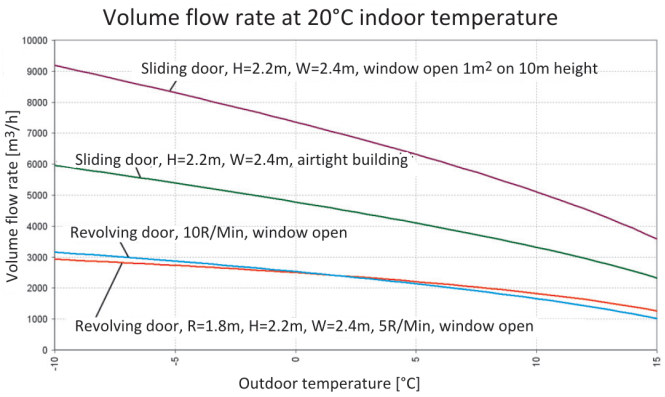

Fig. 2. Volume flow rate of cold air as a function of the temperature difference, from [2].

Lagrangian Coherent Structures: In the seminal work of Haller [15], Lagrangian coherent structures (LCS) were identified as the ridges of the FTLE field. This initiated a considerable amount of research on FTLE itself and FTLE-based visualization. Shadden et al. [33] have shown that FTLE ridges are good approximations of material lines or material surfaces. Therefore, these structures are useful to visualize and analyze transport barriers [19], which is essentially a goal of this paper. LCS have been used to visualize a range of other flow features, such as chaotic structures in vortex rings [34], turbulence [14, 23] and jet streams [37]. Garth et al. presented visualization methods based on FTLE, first for 2D flow [12] and later for 3D flow [10], where, for efficiency reasons, computation was restricted to planar cross sections. Instead of an explicit extraction of ridge surfaces from volumetric FTLE fields, direct volume rendering was used in conjunction with a transfer function that emphasizes high FTLE values. Garth et al. [13] also computed 2D FTLE on offset surfaces of solid boundaries resulting in a visualization of flow separation and flow reattachment. An efficient algorithm for full FTLE computation in 3D was presented by Sadlo et al. [28] using a hierarchical approach where the sampling grid is refined only in the vicinity of ridges. In a later paper [29], temporal coherence of time-dependent velocity data was exploited by a grid advection technique. The same issue was addressed by Lipinski and Mohseni [21], who proposed to track ridges over time to reduce the amount of FTLE computations for time-dependent data. For a more comprehensive overview over FTLE-based visualization, the reader is referred to Section 4 of Pobitzer et al. [26]. As the definition of a ridge in a scalar field is not unique, several different definitions were compared by Schindler et al. [32]. Also, as an alternative way of calculating FTLE ridges, the **C**-ridge [32] was introduced which can be computed without the need of second derivatives from the FTLE field.

3 The Application Case

3.1 Revolving Doors and Air Curtains

The first patent filed on revolving doors was in 1888. Revolving doors were primarily designed to prevent the "entrance of wind, snow, rain or dust" and to improve efficiency because of the "reduced possibility of collision" [6]. Today, they are mainly used to minimize heating effort by preventing air drafts [3]. In a building which is not airtight, a revolving door reduces the draft by roughly a factor of three when compared to a sliding door, and still a factor of two if the building is airtight, see Fig. 2. The prevention of strong air drafts also increases comfort for people working in close proximity such as receptionists. However, revolving doors do not prevent cold air from entering the building, they only prevent air drafts. This is confirmed by a simulation of a revolving door with two wings as seen in Fig. 1(a). Close to the floor, large amounts of cold air is cooling the inside; heating is

needed to compensate this heat loss. A possible solution to this problem is the usage of air curtains. Air curtains are a common measure to create a barrier between the cold air on the outside and the warm air on the inside. They blow warm air typically at rates of 3-7 m/s from the side or from the top, but, depending on the environment, they can also be made stronger. When combined with sliding doors, air curtains have the air outlet and intake arranged in such a way that a roll of warm air or a pair of counter-rotating rolls is generated. This blocks cold outside air and thus reduces draft through the open door, which, in turn, reduces energy loss by approximately 50 percent [31]. For the combination with a revolving door, a different design has been chosen by our application partner: the air curtain blows warm air from the floor, such that it follows the natural convection direction. This leads to a more stable curtain at lower energy consumption. Fig. 1 shows an overview.

3.2 Simulation Data

The simulation of a revolving door with an air curtain was carried out by our industry partners using ANSYS CFX solver [1]. It spans 20 seconds after which a quasi-periodic state is reached, in other words, the flow pattern repeats itself after one period of rotation of the door wings. The temporal resolution of the saved data is 0.1 seconds. The underlying mixed unstructured grid contains approximately 1.65 million cells. The simulation consists of three main parts; the interior, the exterior and the door. The interior and exterior are modeled like a very large room, large enough such that air can flow freely from the door into the in- or outside. The door is 2.5 meters in height and the width of a single door wing is 1.24 meters. The initial goal of the simulation is to test the energy savings when an air curtain is used in combination with a revolving door instead of a sliding door. The simulation does not take into account people going through the door. The revolving door itself has four compartments, which is a very common design for low-traffic doors. The air curtain is idealized by an air outlet with constant velocity and temperature. The air outlet (of the device) is modeled as an inflow boundary of the simulation domain.

4 Lagrangian Coherent Structures

In this section, the necessary background on FTLE and LCS is provided, and several ridge definitions are recapitulated which are important in our context.

4.1 Flow Map and Cauchy-Green Tensor

The *flow map* (in mathematics sometimes simply called the *flow*) of a vector field $\mathbf{u}(\mathbf{x},t)$ is the map from the point where a pathline is seeded at time t_0 to the point where it arrives at time t. The flow map is denoted by $\Phi_{t_0}^{t}(\mathbf{x})$. Formally, it is described by the following initial value problem

$$\frac{\partial}{\partial t}\Phi_{t_0}^{t}(\mathbf{x}) = \mathbf{u}(\Phi_{t_0}^{t}(\mathbf{x}),t), \qquad \Phi_{t_0}^{t_0}(\mathbf{x}) = \mathbf{x}. \tag{1}$$

Its gradient $\mathbf{F} = \nabla\Phi_{t_0}^{t}(\mathbf{x})$ leads to the *(right) Cauchy-Green deformation tensor* $\mathbf{C} = \mathbf{F}^{\top}\mathbf{F}$. This is a positive symmetric tensor, describing the deformation an infinitesimal fluid volume experiences as it is advected up to time t. Its square root $\mathbf{U} = \mathbf{C}^{1/2}$ is the right stretch tensor in the polar decomposition $\mathbf{F} = \mathbf{R}\mathbf{U}$, where $\mathbf{R}$ is a rotation matrix. The eigenvectors $\mathbf{N}_i$ of $\mathbf{U}$ (being also the eigenvectors of $\mathbf{C}$) are orthogonal and contain the directions of maximal and minimal stretch (see Fig. 3).

Given $\mathbf{C}$, the *finite-time Lyapunov exponent (FTLE)* can now be defined as

$$\sigma(\mathbf{x},t_0,t) = \frac{1}{|t-t_0|}\ln\left(\sqrt{\lambda_{\max}(\mathbf{C})}\right) \tag{2}$$

where $\lambda_{\max}(\mathbf{C})$ denotes the largest eigenvalue of $\mathbf{C}$. The FTLE, therefore, describes the rate of separation of a pair of particles, where the maximum is taken over all spatial orientations of the pair. The FTLE can also be computed for a flow map going backward in time. In that case, it describes attraction rather than repulsion.

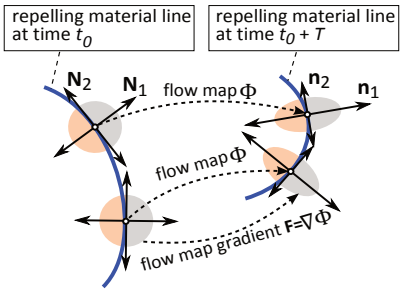

Fig. 3. Tensor field line of $\mathbf{C} = \mathbf{F}^{\top}\mathbf{F}$ acting as a repelling material line. When advected, it is mapped by Φ and their neighborhoods by $\mathbf{F}$.

4.2 Classical Ridge Definitions

Even though ridges in a 2D height field are easily recognized by the human eye, there is not a unique ridge definition. Generally, a ridge is a lower-dimensional manifold in a scalar field that generalizes the notion of a local maximum.

A natural definition of a ridge is the *watershed*. For a 2D scalar field, the watershed is obtained by integrating the gradient of the field, starting at a saddle point. Intuitively, this means walking uphill from a pass to a peak, always taking the direction of the steepest slope. In topological terms, watersheds are part of the topological skeleton of the gradient field. This way, extension to 3D scalar fields is straightforward: They are the unstable 2D manifolds of saddle points. Sahner et al. [30] used such 2D watersheds for their concept of strain skeletons in 3D velocity fields.

Height ridges are a generalization of local maxima. A (maximum convexity) height ridge of co-dimension one is given by the points that have a zero first derivative and a negative second derivative if derivatives are taken in the direction of the minor eigenvector of the Hessian [8]. Here, the minor eigenvector refers to the eigenvector associated with the smallest *signed* eigenvalue. Often this set of points is further reduced by additional constraints in order to remove false positives [20, 25].

Section-based ridges are obtained by finding local maxima in a set of parallel $n-1$-dimensional sections and connecting them [17]. Obviously, the orientation chosen for the sections influences the result, and reasonably good results can be expected only if the ridge is roughly orthogonal to the sections. Section-based ridges have the advantage that they do not require second derivatives.

In 3D velocity fields, it can be observed that the FTLE field often has surface-like maxima, defining structures which are approximately orthogonal to the major eigenvector. Therefore, a height ridge of co-dimension one can be used to represent such structures [28].

4.3 C-Ridges

LCS are obtained by applying any of the above ridge definitions to the FTLE field. Since the FTLE field only depends on the eigenvalues of $\mathbf{C}$, an alternative is to also include the eigenvectors of $\mathbf{C}$. In the concept of the $\mathbf{C}$-ridge [32], the major eigenvector $\mathbf{N}_1$ of $\mathbf{C}$ is used as a predictor for the normal direction of the ridge, with the motivation described in Section 4.1. A ridge point can now be defined as a local FTLE maximum, constrained to the direction $\pm\mathbf{N}_1$. The ridge obtained this way can be expressed by

$$\mathbf{N_1}\cdot\nabla\sigma = 0 \quad \text{and} \quad (\mathbf{H}_\sigma\mathbf{N_1})\cdot\mathbf{N_1} < 0 \tag{3}$$

where $\mathbf{H}_\sigma$ denotes the Hessian of σ. This expression is closely related to the height ridge definition. For the latter, the eigenvector $\mathbf{N_1}$ would have to be replaced by the minor eigenvector of $\mathbf{H}_\sigma$. The $\mathbf{C}$-ridge, if restricted to points where $\sigma > 0$, fulfills two of the conditions of Haller's new concept of *weak LCS* (cf. [16], Definition 7), while the other two conditions (material surface, the surface itself is orthogonal to $\mathbf{N}_1$) can hold in general only in the limit $t \to \infty$.

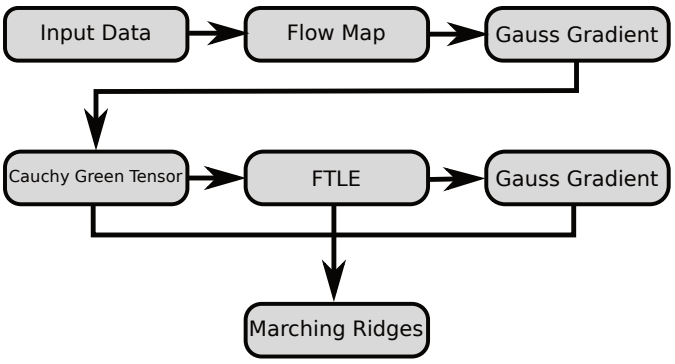

Fig. 4. The visualization pipeline used for computing LCS from a time-dependent velocity field.

5 LCS Visualization Pipeline

In this section the entire ridge extraction pipeline is explained and the algorithmic improvements mentioned in the introduction are explained in detail. An overview of the pipeline is shown in Fig. 4.

First, the flow map, either in forward or backward direction, is computed. Based on the flow map, its gradient is calculated by a convolution with a Gaussian. The Cauchy-Green tensor is then computed and from it, the FTLE field. Since the ridge extraction requires both the FTLE field and the Cauchy-Green tensor, these two operations are computed separately. As ridge extraction depends on the gradient of the scalar field, another Gaussian gradient is computed on the FTLE field. Finally, the ridge extraction is started, having as inputs the eigenvectors of the Cauchy-Green tensor and the gradient of the FTLE field. Since ridges are trimmed at a lower FTLE threshold, the FTLE field is also forwarded to the ridge module.

5.1 Computation of the Flow Map

Visualization by LCS starts with the computation of the flow map. The flow map is sampled on a dense regular grid while the actual integration is done on the original unstructured grid. There are two reasons for this choice: First, the flow map is required in a very high resolution, much finer than the underlying unstructured grid. Second, sampling the flow map on a dense regular grid makes further processing easier as is seen for example in the gradient computation. For the numerical integration, a fourth-order Runge-Kutta method with automatic step size control is used, and velocity data are linearly interpolated between time steps. The computation is parallelized onto all processors/cores of the CPU to speed up the process.

For point location in the unstructured grid, the *cell tree* [11] is used. The cell tree is an adaptive cell location scheme. It is similar to kd-trees, but instead of using a hard boundary to distinguish between the two child nodes, the child nodes can overlap, resulting in a soft boundary. This way, cells do not have to be duplicated in child nodes, saving memory and improving performance significantly.

The data of the four-wing door with an air curtain contains an inflow boundary which models the air curtain. There are air intakes on both sides of the air curtain, as depicted in Fig. 1(d) in red, modeled as outflow boundaries. Our integration implementation has to account for these boundaries, as otherwise, large regions of the domain would fail to reach the required integration time for the flow map. The approach taken was to integrate pathlines across open boundaries, by keeping the velocity vector from the point where the boundary was crossed. Even though this is not a physical extension of the flow field, it is the best approximation that can be done without extending the simulation. Note that this simple scheme is sufficient for our case, but for more complex outflow boundaries, the method by Tang et al. [36] might be more appropriate.

5.2 Computation of FTLE

Given a flow map sampled on a dense regular grid, there are two stages of numerical differentiation involved in the process of first computing an FTLE field and then its ridge surfaces. The first differentiation is needed in the calculation of the Cauchy-Green tensor $\mathbf{C}$, which requires the flow map gradient. In the second stage, the ridge computation, derivatives of the FTLE field are required. Here, second derivatives can be avoided if $\mathbf{C}$-ridges instead of height ridges are used. Nevertheless, the whole process is still numerically highly sensitive. The simplest scheme for estimation of derivatives would be finite differences. However, finite differences do not represent a ground truth, and furthermore, they do not provide good estimates for points between grid nodes. The various numerical methods (based on finite differences, polynomial fitting, or other) all include data from a neighborhood, the size of which determines the amount of smoothing. It is important to have enough smoothing to cancel numerical noise, but not to the point where features start to move. Our solution is the use of Gaussian derivatives, which are obtained by convolving the data with derivatives of a Gaussian. This way, the flow map gradient as well as the smoothed FTLE field, its first and, optionally, second derivatives are reconstructed. The standard deviation of the Gaussian – the *scale* in scale-space terminology – is chosen to be as small as possible, but large enough to avoid discretization artifacts, typically 1.5 to 2.0 times the grid resolution of the flow map. If the kernels are cut off at 4.0 standard deviations, stencils of 15^3 voxels have to be used. This makes convolution an expensive operation that adds significantly to the overall computing time even if separability of the Gaussian kernel is respected.

A much faster way is to make use of the convolution theorem and compute the convolution in frequency space. The Gaussian kernel is separable and therefore the gradient components can be calculated independently. The fast Fourier transform (FFT) and its inverse are available in many different implementations. For our implementation the CUFFT library is used, which is released as part of CUDA [24]. The 3D Gaussian kernel with equal standard deviation across all dimensions centered around the origin is

$$g(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^3 s^3}}\, e^{-\|\mathbf{x}\|^2 / (2s^2)} \tag{4}$$

where s denotes the standard deviation (to avoid confusion with the symbol σ used for the FTLE). Its gradient is then

$$\nabla g(\mathbf{x}) = -\frac{\mathbf{x}}{s^2}\, \frac{1}{\sqrt{(2\pi)^3 s^3}}\, e^{-\|\mathbf{x}\|^2 / (2s^2)} \tag{5}$$

The Fourier transform of the components of a gradient field $\nabla f(\mathbf{x})$ can therefore be computed by component-wise multiplication of $\mathrm{FFT}(f(\mathbf{x}))$ and $\mathrm{FFT}(\nabla g(\mathbf{x}))$.

The Discrete Fourier Transform assumes that the input data is periodic, i.e., the convolution kernel wraps around the data borders. This leads to periodic artifacts at the boundary. To solve this problem, the input data is enlarged in each dimension by the size of the kernel. Frequently, the data is extended using zero-padding. However this would lead to high gradient values at the border which in turn would result in spurious ridges. Therefore, the data is extended using constant propagation, i.e., they have the same value as the closest boundary value. This way, the effect from periodic convolution is suppressed and usable results are achieved at the domain boundaries.

While the convolution-based scheme is generally applicable, its FFT optimization requires that the entire sampling grid is within the domain. "Thin" interior boundaries, such as the door wings and cylinder in our application, can be accepted. Such discontinuities in the gradient estimation lead to artifacts, which can however be effectively eliminated within the ridge extraction step, as is described in the next section.

5.3 Ridge Extraction

For the computation of a **C**-ridge surface, there exist different approaches. One is to use a seed point at a local FTLE maximum and compute the radial extension of the surface from this point [32]. Using the major eigenvector $\mathbf{N}_1$ of **C** as a prediction for the ridge normal, the ridge surface is extended at its current front. The newly added vertices undergo a correction step in the direction of the local eigenvector $\mathbf{N}_1$, maximizing the FTLE value σ. Integration is stopped at vertices where σ drops below a given threshold.

Such a seed based method is especially useful for extracting a single ridge and tracking it over time. The alternative approach, which is our choice, is based on the *Marching Ridges* algorithm [9]. It can generate both height ridges and **C**-ridges.

Before the Marching Ridges procedure starts, the tensor **C** and the FTLE value σ are computed on each grid node, as is described in Fig. 5. Also, depending on the chosen ridge type, one or two derivatives of σ are computed. Finally, an eigenvector **e** is computed per grid node, which is $\mathbf{N_1}$ in the case of a **C**-ridge and the minor eigenvector of $\mathbf{H}_\sigma$ in the case of a height ridge.

Now the *Marching Ridges* algorithm starts looping over all grid cells. A principal components analysis (PCA) is performed on the eigenvectors (and their negatives) at the eight nodes of the hexahedral cell. The first principal component is used to consistently orient the eight eigenvectors of the tensor. These are then used to compute the scalar product $\mathbf{e} \cdot \nabla\sigma$ corresponding to the first part of Eq. 3. From these scalar values at the eight nodes, the zero isosurface is computed with the standard Marching Cubes [22] method. This results in a set of triangles the vertices of which satisfy the first part of Eq. 3, but which may be valley points instead of ridge points. That means that some vertices, together with parts of the triangles, have to be trimmed. In order to decide this for a vertex at position **x**, the first and second derivatives of σ are reconstructed at **x** and the second part of Eq. 3 is evaluated. Here is the only place where second derivatives of σ are needed in the case of **C**-ridges. We avoid them by using two test points $\mathbf{x} \pm h\mathbf{e}$ where h is the grid resolution and **e** is the eigenvector reconstructed at **x**. A local maximum can be checked by reconstructing σ at these test points and comparing it with the value at **x**.

The second reason for trimming is an FTLE value falling below a given threshold σ_{min}. If only part of a triangle has to be trimmed, the trimming line is the isocontour at level σ_{min}. Given the three reconstructed σ values, there are five different trimming cases to consider, as seen in Fig. 6.

The ridge extraction process assumes that the FTLE gradient is tangential to the extracted surface. However, it has been shown [25] that the gradient can significantly deviate from the surface tangent. This happens in areas where the ridge is not pronounced. It therefore makes sense to remove such parts from the output mesh to reduce clutter and to make the output more meaningful. By construction, the major eigenvector of the Cauchy-Green tensor is orthogonal to the FTLE gradient at the mesh vertices. We therefore use the angle between the averaged mesh normals and this eigenvector. An example showing the effectiveness of this trimming can be seen in Fig. 7.

The described algorithm is in most parts the same for **C**-ridges and height ridges. A major difference is that height ridges require second derivatives of σ already in the computation of ridge vertices, not just in the subsequent trimming of the ridge surface. This is the reason why height ridges are more expensive to compute and typically give noisier results than **C**-ridges.

One problem of the approach used here is that the use of a dense regular grid disregards the building geometry. To solve this problem, we introduce a secondary array allowing us to mark individual grid cells as invalid. All cells intersecting with the building geometry are marked as invalid. The ridge extraction algorithm then respects this information by only considering cells which are not marked as invalid. Naturally, vertices of cells containing geometry will have high FTLE values due to the different flow behavior of the areas separated by the geometry. As a result, ridges would be extracted from these cells if they were not skipped. Although this solution is not mathematically rigorous, experience has shown that it very effectively eliminates spurious ridges.

```
Triangles RidgeSurface(FlowMap Φ, bool ridgeMode)
// Input: flowmap sampled on a regular grid.
// Output: non-oriented triangles

// 1. Compute flow map gradient
Φ̂ ← FFT(Φ)
foreach grid node do
    F̂ ← FrequencyDomainGradient(Φ̂)
F ← InverseFFT(F̂)
// 2. Compute C and FTLE (σ)
foreach grid node do
    C ← F^⊤ · F
    σ ← FTLE(C)
// 3. Compute FTLE gradient and Hessian if required
σ̂ ← FFT(σ)
foreach grid node do
    ∇̂σ ← FrequencyDomainGradient(σ̂)
    if (ridgeMode) then Ĥ_σ ← FrequencyDomainHessian(σ̂)
∇σ ← IFFT(∇̂σ)
if (ridgeMode) then H_σ ← IFFT(Ĥ_σ)
// 4. Calculate eigenvectors
foreach grid node do
    if (ridgeMode) then e ← MinorEigenvector(H_σ)
    else e ← MajorEigenvector(C)
// 5. Extract Ridge
foreach grid cell do
    N ← set of eight adjacent nodes
    E ← set of eight eigenvectors e at nodes in N
    p ← FirstVector(PrincipalComponentAnalysis(E))
    // 5.1 Orient eigenvectors consistently using PCA
    foreach e in E
        if e · p < 0 then e ← −e
        s ← e · ∇σ

    (V,T) ← MarchingCubesVerticesAndTriangles(s, level = 0)

    // 5.2 Classify vertex (ridge or valley)
    foreach vertex v in V do
        H_σ ← spatialDomainHessian(Φ, v)
        (σ, e) ← interpolationOnEdge(σ, e, v)
        d ← (H_σ e) · e    // negative for ridge, positive for valley

    // 5.3 Trimming
    foreach edge of T do
        if σ < σ_low at both endpoints then remove edge
        if σ < σ_low at one endpoint then trim edge at σ = σ_low
        if d > 0 at both endpoints then remove edge
        if d > 0 at one endpoint then trim edge at d = 0

    Generate output triangles, using shared vertices
```

Fig. 5. Ridge surface algorithm. FTLE is defined by equation 2. IFFT is an abbreviation for Inverse FFT.

5.4 Non-Orientability of Ridge Surfaces

A problem arises in the process of rendering ridge surfaces. A triangle with vertices v_1, v_2, v_3 has a *positive face*, on which the vertices appear in counter-clockwise order, and a *negative face*, on which they appear in clockwise order. Since ridges are based on eigenvectors (of either the Hessian or the Cauchy-Green tensor), the resulting triangle mesh will not have consistently oriented triangles. A possible solution to this problem is traversing the mesh using breadth-first search and flipping triangles that are not consistent. A problem occurs when the

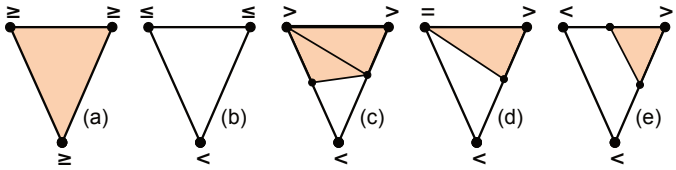

Fig. 6. Different trimming cases, based on the three reconstructed values of σ and σ_{min}. If all values are above or equal to σ_{min}, the triangle is kept (a). If all are lower, the entire triangle is discarded (b). Otherwise, the triangle is trimmed accordingly (c-e).

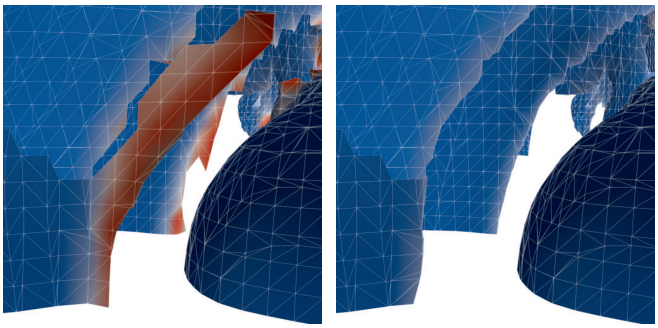

Fig. 7. Closeup of an extracted ridge without (left) and with (right) angle-based trimming. The mesh is colored based on the angle (α) between the vertex normal and the eigenvector of the Cauchy-Green tensor where blue denotes $\alpha = 0$ and red denotes $\alpha = \pi/2$

mesh is not orientable, e.g., has the form of a Möbius strip. The border between differently oriented triangles can, however, be easily detected by the described breadth-first search. The mesh is then cut at this border, which results in a consistently oriented mesh. The vertex normals for border vertices can be correctly computed by taking all incident triangles and their orientation into account. The resulting mesh can then be rendered without artifacts using two-sided lighting.

5.5 Complementary Visualization

Along with the LCS, several basic visualization techniques complement the analysis of the air flow. Slicing planes or isosurfaces of the temperature help to understand the temperature distribution. The temperature certainly is the most fundamental field to look at for the engineers, however, it is important to realize that temperature alone does not provide the information contained in LCS. Because the flow is not caused by convection alone, but actively influenced by the door and the air curtain, temperature contours and LCS are not necessarily aligned. This can be seen best inside the sectors of the revolving door, where the temperatures form layers that are roughly horizontal, while LCS are not aligned with these layers. In other words, a picture of the temperature distribution does not explain what caused it. Therefore, the velocity field, which is a product of convection and the door's pumping effect, has to be studied. Its structures are not necessarily in agreement with high-temperature gradients. LCS can give valuable insight into the structure of the flow, however, complementary information is usually needed because LCS do not provide important information such as the current flow direction. Therefore, we also use pathlines and particle advection. Moreover, these techniques are also used to analyze the flow behavior at places where no ridges are present. To give the viewer a feeling for the surroundings, a reduced version of the door geometry is also rendered transparently.

6 Discussion / Results

In this section, we discuss our analysis of the air flow using LCS. We start by inspecting the temperature distribution and highlight its limitations. Then, we extract the basic topological structure of the airflow in its quasi-periodic state. We show how the air curtain fails to fully shield off the interior by exposing a small heat leak. Two

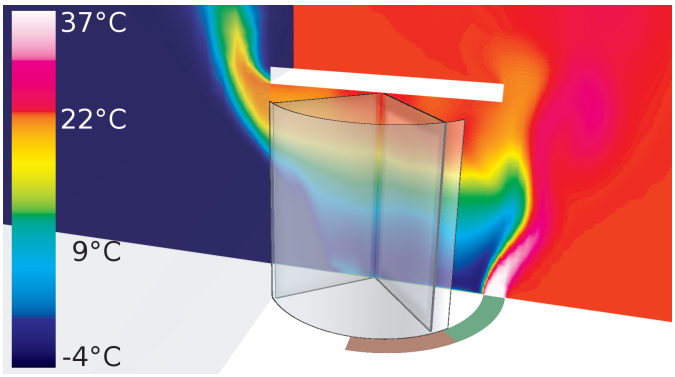

Fig. 8. The temperature slice at t = 12.35 shows that the air curtain manages to hold off the cold air. But it is also evident that no information about the flow structure is gained this way.

alternative scenarios are then studied, which attempt to close this leak. Finally, we provide performance numbers showing that the **C**-ridges can be extracted fast enough to be usable by engineers in practice. All techniques listed here were implemented in the Visdom visualization framework [38].

6.1 Temperature Distribution

The temperature distribution is naturally the first thing the engineer will look at when simulating such a scenario. For that, we chose to sample the temperature on a plane, as is shown in Fig. 8. As can be clearly seen, the cold air at the bottom of the door is held off very well by the air curtain. This is an important first insight, confirming the initial idea of combining air curtains and revolving doors. Also the significant amount of warm air escaping the doorway is visible. An air curtain blowing cold air from the top would help here, however it is not clear whether the gain in efficiency would outweigh the effort of another curtain. Also, according to our domain expert, legal requirements can prohibit such a design.

The visualization of the temperature distribution gave a first helpful overview of the simulation. However, it is also clear that there is little information gained on the air flow structure. For example, the origin of the warm air exiting the building can only be guessed. The air curtain as well as the interior are both potential sources.

6.2 General Overview Using LCS-Based Visualization

A rendering of the **C**-ridges at the beginning of one period is depicted in Fig. 9. Attracting LCS are colored in red, while repelling LCS are colored in blue. The rendering uses adaptive transparency and a dense flow visualization [4] to make it easier to understand the structure of the LCS. We focus on the area around the air curtain and ignore the ridges originating from warm air escaping to the exterior. There are three main structures here: the large attracting LCS originating directly from the air curtain marked (1), the periodically occurring "shovels" (repelling LCS) moving upwards (2), and the bubbles at the floor (3). For illustration purposes and because interpreting LCS is not easy, we look at cross sections of the ridges for illustration purposes as seen in Fig. 10 and Fig. 11. Of central importance for the air curtain is the blue repelling ridge labeled (1) in Fig. 10. Air on the left of that ridge remains within the door, prevented from entering the interior. On the other side, the air is pulled up by the air curtain, warmed up and, depending on the time within the period, it is brought back into the door or released into the interior of the building. The temperature of the air curtain manages to heat up this air so that no temperature drops are visible. This observation is confirmed by the particle animation in the accompanying movie and by the temperature slice animation. Going back to the 3D visualization in Fig. 9, we quickly see that this separation not only works in the middle of the entrance, but across the entire opening.

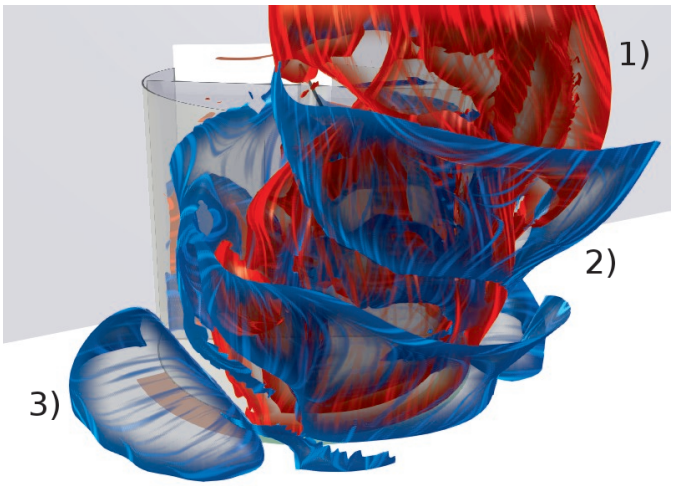

Fig. 9. General overview of the ridge structure. Red and blue surfaces depict attracting and repelling LCS, respectively. Ridges were trimmed at an FTLE threshold of 1.5. The large attracting structure marked (1) originates from the air curtain. The repelling structures (2) originate from circulating structures in ascending air. The repelling structure (3) can be seen as a simulation artifact originating from the idealized air intake.

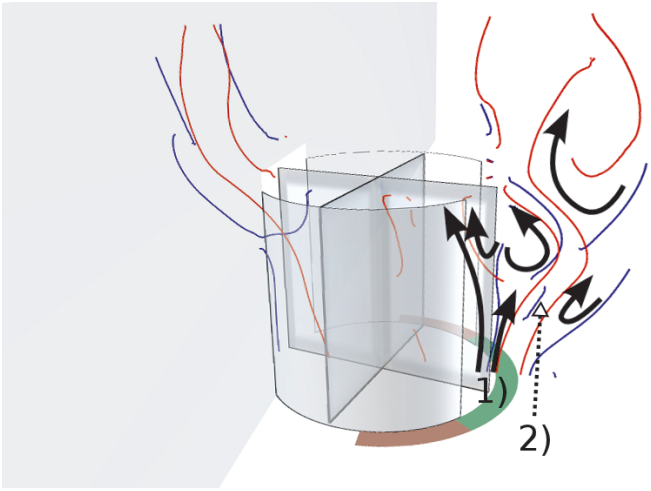

Fig. 10. Cross section of the LCS showing the flow structure in the symmetry plane of the door. From repelling (blue) and attracting (red) LCS the motion of the flow relative to the moving LCS can be inferred. 1) The four arrows on the left show the flow barrier created by the air curtain. 2) The two arrows on the right show circulation within the ascending warm air.

The repelling LCS marked (2) in Fig. 10 tends to break apart within a period, as seen in Fig. 11. It separates air from the air curtain which is released into the door from air released into the interior of the building. If it would not break apart, particles left of this ridge would always end up within the door area, which has been shown not to be the case in the particle animation.

6.3 Heat Leak

Even though the ridges indicate a near optimal separation, the provided visualizations suggest that more improvements can be made to the design in order to further optimize energy saving. In Fig. 12 a close-up of the ridges near the door as well as four pathlines are shown. The absence of ridges in the area between the curtain ridges and the revolving door is notable. This indicates that no shielding takes place there and that the flow behavior should be investigated in more detail. We therefore seeded a few path lines in the interior close to this area. These pathlines move directly into one of the sectors of the revolving door. This is caused by the pressure gradient resulting from the rotating wings. As a result, warm air from the interior gets pulled out, causing heat leakage.

We were surprised to find that we were most interested in areas with an unexpected lack of LCS. LCS, or the lack thereof, really proved their usefulness here as they made it possible to find and localize such a heat leak efficiently. Even though we used pathlines to confirm our finding, using only pathlines would have forced the engineer to search the entire domain.

6.4 Fixing the Heat Leak

Our results were met with great enthusiasm by our industry partners. As a result, our domain experts reran the same simulation with two small modifications in an attempt to fix the described heat leak. The changes are highlighted in Fig. 13. In the first case, the air curtain is extended at the left side in order to better shield off the leak. In the second case, a small flap is added to the top of the door entrance with the idea of blocking air entering the door due to the pumping effect.

The results can be seen in Fig. 14 for the first case, in Fig. 15 for the second case. In the first case, the LCS nicely shield the entire door area. In both cases, the size of the leak is minimized greatly, meaning that the amount of air sucked into the doorway is much smaller.

Our LCS-based visualizations led to two design modifications, both of which eliminate the largest heat leak. Whether the modified designs

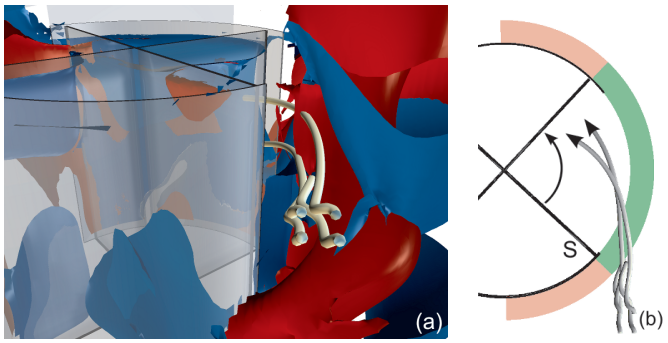

Fig. 12. (a) The air curtain fails to seal off the interior completely. Warm air behind the door can exit through the door, as indicated by an area of missing LCS. Pathlines confirm this behavior. The reason why some pathlines cross the attracting LCS is that pathlines depict an entire time interval, during which the LCS move. Here, LCS are shown at the time of pathline seeding. (b) The top view shows the door in its position at the seeding time. Particles enter the door sector at the suction side of the wing marked with "S". Images are taken at time step $t = 15.1$.

also lead to energy saving, is not yet clear. A calculation of the total volume flux, integrated over a whole period, shows only a small effect within the band of fluctuations. While a refined analysis will be needed, it can be stated that our visualizations not only helped understanding the air flow but can actively influence design decisions for the building of energy-efficient doors.

6.5 Performance Analysis

All performance measures were carried out on a 2x Quad Core Xeon E5430 machine with 64GB of RAM running Linux. The GPU used is an NVIDIA GTX 580.

6.5.1 Flow Map Computation

The computation of the flow map is by far the most expensive part for any FTLE-based visualization. Therefore, the computed flow map was saved to disk to facilitate further repeated processing. However, as a result of this, the integration time is fixed for the entire analysis. The integration time was chosen to be three seconds which is slightly more than one period of the simulation. Since one period consists of the door only rotating by 90 degrees, periodic artifacts are not expected. The

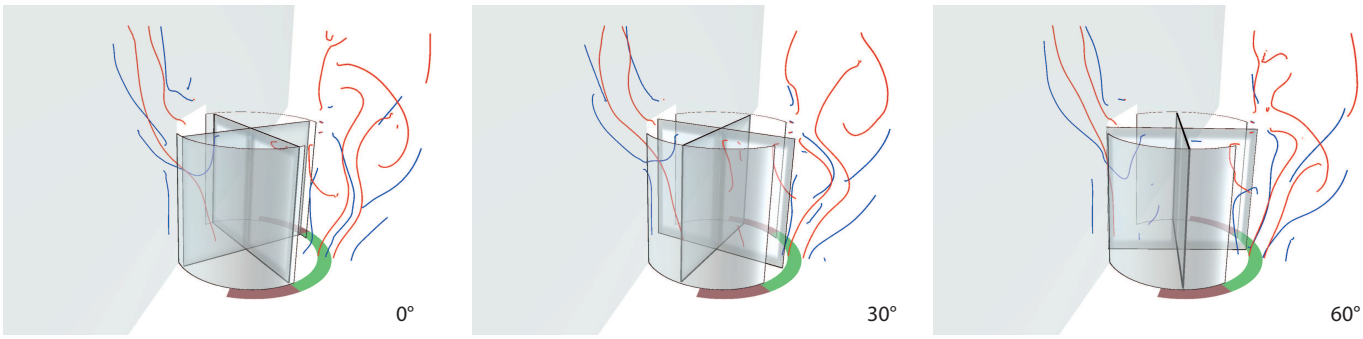

Fig. 11. Sliced view of the LCS over the time span of the door rotating 90 degrees (one period). Pairs of repelling (blue) and attracting (red) LCS form flow barriers over the full height, which separate air flows in the door and in the interior. The LCS marked * are within the air curtain, indicating that parts of the warm air enter the door area shielding the inside and warming up cold air coming from the outside. The LCS marked B splits into two parts between 30 and 60 degrees.

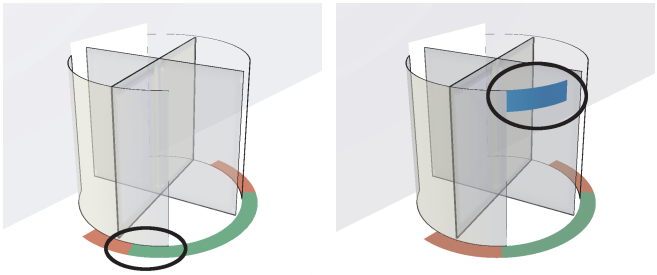

Fig. 13. The additional simulations performed in order to minimize the heat leak. The highlighted areas denote the modifications done to the original case. *Left:* The air curtain is extended on the left side. *Right:* A small flap depicted in blue is added in order to avoid the leak.

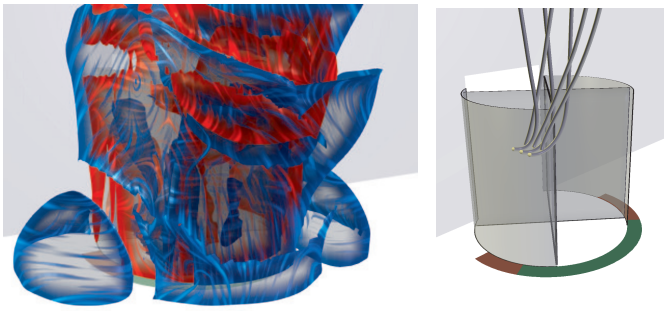

Fig. 14. Results for the first modification case (longer air curtain). *Left:* the LCS cover the entire door, leaving only a small gap as opposed to the original case as seen in Fig. 12 (a). *Right:* pathlines seeded as in the original case (Fig. 12) are diverted and stay in the interior.

computation of the flow map is carried out directly on the unstructured grid (without any resampling). The flow map is sampled on a regular grid with dimensions $140 \times 120 \times 80$. These numbers were chosen to prevent flow map computation in uninteresting regions while retaining equal spacing in all three dimensions. Also, it should be pointed out that no adaptive computation of the flow map [29] was used.

The computation of one single flow map takes about 4 minutes, resulting in 8 minutes of computation per time step when also counting the backwards flow map. This resulted in a total of 37 hours of computation for all of the 280 time steps. Because the flow map for the entire data is precomputed, this rather large amount of time does not impact the data analysis process by engineers.

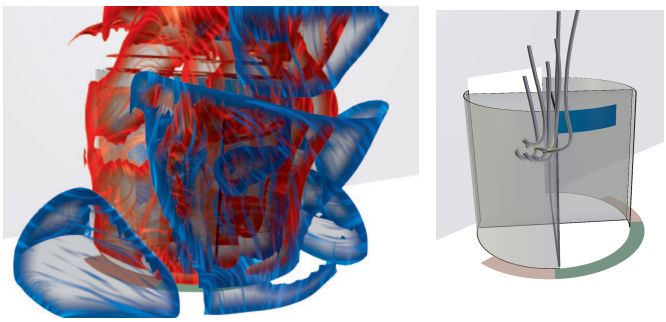

Fig. 15. Results for the second modification case. *Left:* The repelling LCS (blue) closely follows the door border. There is a hole in the red attracting ridge, however it does not extend to the entrance. The pathlines are also an indication of this.

Step	Time (ms)
Flow map gradient	224
Cauchy-Green tensor	461
FTLE computation	653
FTLE gradient	119
Ridge extraction	1324
Total	2781

Table 1. Performance numbers for the steps of the visualization pipeline.

6.5.2 Ridge Extraction

C-Ridges are extracted using the FTLE gradient and the major eigenvector of the Cauchy-Green tensor, as seen in Fig. 4. The performance numbers for every step in the pipeline can be seen in Table 1. The difference in performance between FTLE gradient and the flow map gradient is explained by the fact that the flow map gradient requires the differentiation of three components while the FTLE field is simply a scalar.

The total extraction time from loading of the flow map to the ridge geometry is in the area of six seconds (when counting both forward and backwards ridges), which is more than fast enough for practical application.

6.5.3 FFT-Based Gradient vs. Explicit Convolution

We compare the performance of our FFT-based gradient implementation to our initial convolution-based implementation. The FFT-based version is implemented on the GPU using CUFFT while the convolution-based version is single-threaded CPU code. The FFT based version is of course expected to be faster as it has an asymptotically better runtime than direct convolution ($O(n \log n)$ vs. $O(nm)$). Even though we are aware that comparing such vastly different ap-

proaches is questionable, the output of both algorithms is the same and more importantly, they took a similar amount of time to implement.

With variance set to 2, the FFT-based implementation took 224ms as seen in Table 1. In comparison, the convolution-based code took 6 minutes and 42 seconds. This enormous boost in performance is what makes on-the-fly extraction of ridges possible without any preprocessing other than the flow map. Also, by storing the flow map on disk, the least amount of data is saved as only three floats per grid element are written on disk.

7 Conclusion and Future Work

In this paper, we have shown how topological visualization techniques helped understanding the flow around a revolving door when an air curtain is added. Based on the analysis, two additional simulations were carried out to optimize the application at hand.

Insight: The extracted LCS ridges helped in getting a deep understanding of the flow structure around the air curtain, which would not have been possible by looking at temperature plots alone. Foremost, the results obtained led to the conclusion that the air curtain is capable of holding off cold air from entering the building over a large fraction of the area of the door. The LCS ridges revealed heat leaks above the air curtain which we then verified interactively using pathlines. These results were used to further improve the design, significantly reducing the size of the mentioned heat leak.

Algorithmic Improvements: On the algorithmic side, this paper proposes several improvements. Foremost, the FFT was used to calculate spatial gradients for both the flow map and the FTLE field, resulting in significant speed-ups. Boundary trimming of ridges helps to get smooth ridge boundaries and to achieve high-quality renderings. Finally, the non-orientable ridge mesh was made orientable by cutting it in pieces, while keeping surface normals consistent.

Limitations: The LCS-based visualizations tend to be cluttered. We solved this by using an illustrative visualization technique with adaptive transparency and by looking at cross sections. Still, the resulting visualizations take time to understand, and interactive viewing is advantageous. Also, change in air flow by people walking through the door is not simulated. On the algorithmic side, the FFT-based gradient estimation scheme ignores simulation geometry as it is based on a regular grid. Even though we were able to filter out artifacts effectively, full inclusion of the simulation geometry would be desirable.

Future Work: As future work, looking for ways to speed up the flow map computation is an obvious goal as it is computationally by far the most expensive part. Also, depicting the flow behavior from the FTLE ridges requires some experience because such visualizations focus on the semantics of the flow rather than its immediate structure. Therefore, visualization techniques which intuitively highlight the flow behavior in volumes separated by the LCS would be of benefit.

Acknowledgments

We thank Matthias Röthlin for computing the CFD simulations. This work was partially funded by the Swiss National Science Foundation under grant 200021_127022 and in part by a grant from the Austrian Science Fund (FWF):P 22542-N23 (Semantic Steering). The project SemSeg acknowledges the financial support of the Future and Emerging Technologies (FET) programme within the Seventh Framework Programme for Research of the European Commission, under FET-Open grant number 226042.

References

[1] ANSYS, Inc. ANSYS CFX, Release 12.1. `http://www.ansys.com/`, 2009.

[2] S. Barp. Wenn Luftströme zum Problem werden. *Haustech Extra*, 234(3):51–52, 2010.

[3] A. Beardmore. *The revolving door since 1881: Architecture in detail*. Boon Edam, 2000.

[4] R. Carnecky, B. Schindler, R. Fuchs, and R. Peikert. Multi-layer illustrative dense flow visualization. *Computer Graphics Forum*, 31(3):895–904, 2012.

[5] M. Cehlin. *Visualization of Air Flow, Temperature and Concentration Indoors : Whole-field measuring methods and CFD*. PhD Dissertation, KTH Stockholm, 2006.

[6] B. Cullum, O. Lee, S. Sukkasi, and D. Wesolowski. Modifying Habits Towards Sustainability: A Study of Revolving Door Usage on the MIT Campus. Technical report, MIT, 2006.

[7] J. P. den Hartog. *Designing Indoor Climate: A Thesis on the Integration of Indoor Climate Analysis in Architectural Design*. PhD thesis, TU Delft, 2003.

[8] D. Eberly. *Ridges in Image and Data Analysis*. Kluwer Acadamic Publishers, 1996.

[9] J. D. Furst and S. M. Pizer. Marching Ridges. In *Proceedings of the IASTED International Conference on Signal and Image Processing*, pages 22–26, 2001.

[10] C. Garth, F. Gerhardt, X. Tricoche, and H. Hagen. Efficient Computation and Visualization of Coherent Structures in Fluid Flow Applications. *Transactions on Visualization and Computer Graphics*, 13(6):1464–1471, 2007.

[11] C. Garth and K. Joy. Fast, Memory-Efficient Cell Location in Unstructured Grids for Visualization. *Transactions on Visualization and Computer Graphics*, 16(6):1541–1550, 2010.

[12] C. Garth, G.-S. Li, X. Tricoche, C. D. Hansen, and H. Hagen. Visualization of Coherent Structures in Transient 2D Flows. In *Topology-Based Methods in Visualization II*, pages 1–13, 2009.

[13] C. Garth, A. Wiebel, X. Tricoche, K. I. Joy, and G. Scheuermann. Lagrangian Visualization of Flow-Embedded Surface Structures. *Computer Graphics Forum*, 27(3):1007–1014, 2008.

[14] M. A. Green, C. W. Rowley, and G. Haller. Detection of Lagrangian coherent structures in three-dimensional turbulence. *Journal of Fluid Mechanics*, 572:111–120, 2007.

[15] G. Haller. Distinguished Material Surfaces and Coherent Structures in Three-Dimensional Fluid Flows. *Physica D*, 149(4):248–277, 2001.

[16] G. Haller. A Variational Theory of Hyperbolic Lagrangian Coherent Structures. *Physica D*, 240(7):574–598, 2010.

[17] E. Johnston and A. Rosenfeld. Digital detection of pits, peaks, ridges, and ravines. *IEEE Transactions on Systems, Man, and Cybernetics*, 5(6):472–480, 1975.

[18] S. Kenjeres, S. B. Gunarjo, and K. Hanjalic. Visualization of air flow and smoke spreading for realistic indoor-climate situations. *Journal of Visualization*, 7(4):268, 2004.

[19] F. Lekien, S. C. Shadden, and J. E. Marsden. Lagrangian coherent structures in n-dimensional systems. *Journal of Mathematical Physics*, 48(6):065404.1–19, 2007.

[20] T. Lindeberg. Edge detection and ridge detection with automatic scale selection. *International Journal of Computer Vision*, 30(2):117–154, 1998.

[21] D. Lipinski and K. Mohseni. A ridge tracking algorithm and error estimate for efficient computation of Lagrangian coherent structures. *Chaos*, 20(1):017504.1–9, 2010.

[22] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. *Computer Graphics*, 21(4):163–169, 1987.

[23] M. Mathur, G. Haller, T. Peacock, J. E. Ruppert-Felsot, and H. L. Swinney. Uncovering the Lagrangian Skeleton of Turbulence. *Physical Review Letters*, 98(14):144502.1–13, 2007.

[24] NVIDIA. CUDA Compute Unified Device Architecture. `http://developer.nvidia.com/object/gpucomputing.html` (last visited Mar. 20, 2012).

[25] R. Peikert and F. Sadlo. Height Ridge Computation and Filtering for Visualization. In *Proceedings of Pacific Vis 2008*, pages 119–126, 2008.

[26] A. Pobitzer, R. Peikert, R. Fuchs, B. Schindler, A. Kuhn, H. Theisel, K. Matković, and H. Hauser. The State of the Art in Topology-Based Visualization of Unsteady Flow. *Computer Graphics Forum*, 30(6):1789–1811, 2011.

[27] F. Reinders, F. Post, and H. Spoelder. Attribute-based Feature Tracking. In *Data Visualization 1999*, pages 63–72, 1999.

[28] F. Sadlo and R. Peikert. Efficient Visualization of Lagrangian Coherent Structures by Filtered AMR Ridge Extraction. *Transactions on Visualization and Computer Graphics*, 13(6):1456–1463, 2007.

[29] F. Sadlo, A. Rigazzi, and R. Peikert. Time-Dependent Visualization of Lagrangian Coherent Structures by Grid Advection. In *Topological Data Analysis and Visualization*, pages 151–165, 2011.

[30] J. Sahner, T. Weinkauf, N. Teuber, and H.-C. Hege. Vortex and Strain Skeletons in Eulerian and Lagrangian Frames. *Transactions on Visual-*

ization and Computer Graphics, 13(5):980–990, 2007.

[31] A. Schälin and J. Martinek. Luftschleiersysteme für Gebäudeeingänge. Technical Report BK_1999-08-01, AFC Bauklimatik, AFC Air Flow Consulting, Zürich, 1999.

[32] B. Schindler, R. Peikert, R. Fuchs, and H. Theisel. Ridge Concepts for the Visualization of Lagrangian Coherent Structures. In R. Peikert, H. Hauser, H. Carr, and R. Fuchs, editors, *Topological Methods in Data Analysis and Visualization II*, pages 221–236. Springer, 2012.

[33] S. Shadden, F. Lekien, and J. Marsden. Definition and properties of Lagrangian coherent structures from finite-time Lyapunov exponents in two-dimensional aperiodic flows. *Physica D*, 212:271–304, 2005.

[34] S. C. Shadden, J. O. Dabiri, and J. E. Marsden. Lagrangian analysis of fluid transport in empirical vortex ring flows. *Physics of Fluids*, 18(4):047105.1–11, 2006.

[35] O. Staubli, C. Sigg, R. Peikert, M. Gross, and D. Gubler. Volume rendering of smoke propagation CFD data. In *Proceedings IEEE Visualization 2005*, pages 335–342, 2005.

[36] W. Tang, P. W. Chan, and G. Haller. Accurate extraction of Lagrangian coherent structures over finite domains with application to flight data analysis over Hong Kong International Airport. *Chaos*, 20(1):017502,1–8, 2010.

[37] W. Tang, M. Mathur, G. Haller, D. C. Hahn, and F. H. Ruggiero. Lagrangian Coherent Structures near a Subtropical Jet Stream. *Journal of the Atmospheric Sciences*, 67:2307–2319, 2010.

[38] J. Waser, R. Fuchs, H. Ribicic, B. Schindler, G. Blöschl, and E. Gröller. World Lines. *Transactions on Visualization and Computer Graphics*, 16(6):1458–1467, 2010.