pp 547–571. © Royal Aeronautical Society 2016. This is an Open Access article, distributed under the terms of the Creative Commons Attribution licence (http://creativecommons.org/licenses/by/4.0/), which permits unrestricted re-use, distribution, and reproduction in any medium, provided the original work is properly cited.
doi: 10.1017/aer.2016.10

Enhanced flow visualisation of complex aerodynamic phenomena using automatic stream surface seeding with application to the BLOODHOUND SSC Land Speed Record vehicle

M. Edmunds
m.edmunds@swansea.ac.uk

B. Evans and I. Masters
Zienkiewicz Centre for Computational Engineering
College of Engineering
Swansea University
UK

R. S. Laramee
Visualisation Group
Computer Science Department
Swansea University
UK

ABSTRACT

This application paper describes a novel, cluster-based, semi-automatic, stream surface placement strategy for structured and unstructured computational fluid dynamics (CFD) data, tailored towards a specific application: The BLOODHOUND jet and rocket propelled land speed record vehicle. An existing automatic stream surface placement algorithm[8], is extensively modified to cater for large unstructured CFD simulation data. The existing algorithm uses hierarchical clustering of velocity and distance vectors to find potential stream surface seeding locations. This work replaces the hierarchical clustering algorithm, designed to work with small regular grids, with a K-means clustering approach suitable for large unstructured grids. Modifications are made to the seeding curve construction algorithm, improving the smoothness and distribution of the discretised curve in complex cases. A new distance function is described which allows the user to target particular characteristics

Received 5 March 2015; revised 24 September 2015; accepted 24 September 2015.

of simulation data. The proposed algorithm reduces the required memory footprint and computational requirement compared to previous work[8]. The performance and effectiveness of the proposed algorithm is demonstrated, and CFD domain expert evaluation is provided describing the value of this approach.

Keywords: CFD; flow visualisation; BLOODHOUND SSC; aerodynamics

NOMENCLATURE

LSR	land speed record
CFD	computational fluid dynamics
SSC	super-sonic car
$\mathbf{p}$	point in 3D space
$\mathbf{v}$	velocity vector
$\mathbf{c}$	curvature vector
$\mathbf{g}$	velocity gradient vector
$\mathbf{e}$	Euclidean distance vector
∇	gradient operator
$\lvert\mathbf{x}\rvert$	magnitude $\mathbf{x}$
$\hat{\mathbf{x}}$	range normalised $\mathbf{x}$
i, j, k	unit vector components
$A\ B$	algorithm bias parameters
k	quantity of clusters
π	cluster partition
c	cluster centroid
s	scalar attribute
O	complexity
n	number of samples
ρ	density
c_p	coefficient of pressure
E	energy
Ω	unstructured tetrahedral mesh

1.0 INTRODUCTION

1.1 Background and motivation

In recent decades, aerodynamic designers, and design engineers more generally, have increasingly relied on computational modelling, and in particular computational fluid dynamics (CFD) to simulate the aerodynamic response of motorsport[20] and aerospace vehicles. A typical multi-disciplinary aerospace design cycle is shown in Fig. 1, and it is evident that CFD analysis and CFD post-processing lie at the heart of the design cycle's 'inner loop'. The impact of this is that the ability of a designer to interrogate the results from CFD analyses in a meaningful and insightful manner is crucial for effective and efficient design iterations. Typically an aerodynamicist is interested in the force coefficient (e.g. lift and drag coefficient) response to any changes in the external shape of a vehicle and seeks to understand the physical phenomena that underlie the changes in these force coefficients

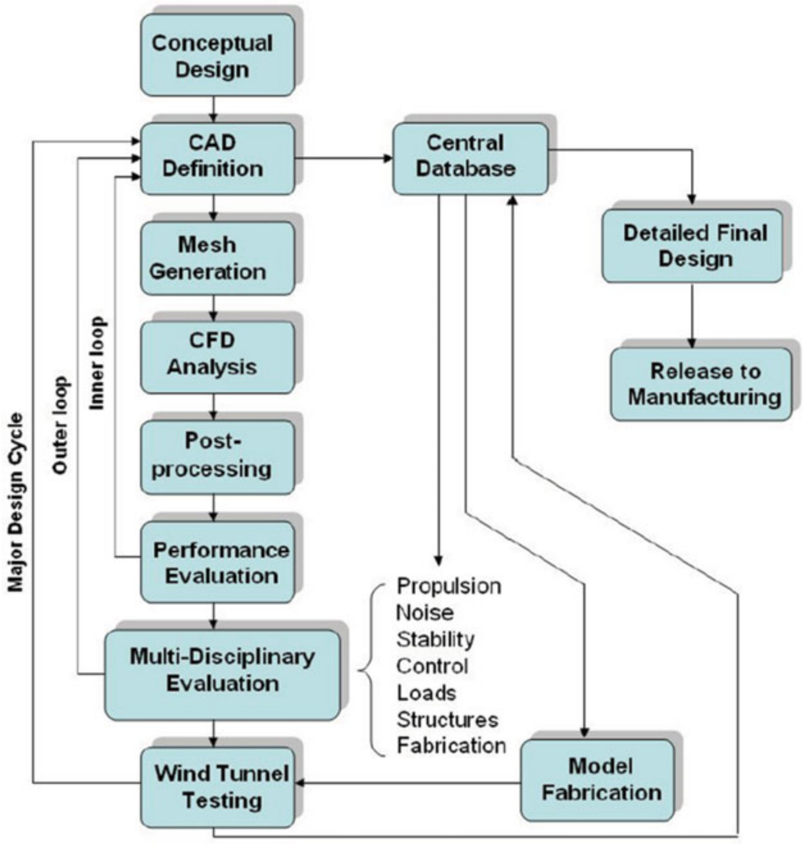

Figure 1. (Colour online) A typical multi-disciplinary aerospace design cycle – note the importance of CFD simulation and post-processing at the heart of the cycle.

using flow visualisation. Very often this is done using commercial (or open source) post-processing or data visualisation software such as EnSight(11) or Paraview(27). The techniques and applications presented in this paper are the results of work undertaken to improve on the effectiveness of flow visualisation in the context of the typical aerodynamic design cycle. The specific application that the approaches detailed in this paper were developed to tackle was the aerodynamic design of BLOODHOUND SSC. BLOODHOUND SSC is a British jet and rocket-powered car designed between 2007 and 2014 with the objective of taking the land speed record (LSR) to 1000 mph (approximately Mach 1.3)(12). The BLOODHOUND vehicle has four wheels and will be under full control of its driver during record attempts set to take place in 2015 and 2016. It has a slender body of approximately 13 m length with two front wheels within the body and two rear wheels mounted externally within wheel fairings. It weighs over 7 tonnes and the engines generate more than 135,000 horsepower. BLOODHOUND has been developed utilising a mixture of motorsport and aerospace technology. The front half of the car is a carbon fibre monocoque and the rear half is constructed as a metallic spaceframe(26). An artist's impression of the final vehicle is shown in Fig. 2.

Figure 3 shows a predicted 1000 mph record attempt run profile. Note that a significant aerodynamic challenge for this vehicle is the design of the airbrakes that will be used to slow

Figure 2. (Colour online) Artist's impression of BLOODHOUND SSC. At the time of writing, the vehicle is under construction with completion due in 2015 and testing scheduled for 2015 and 2016.

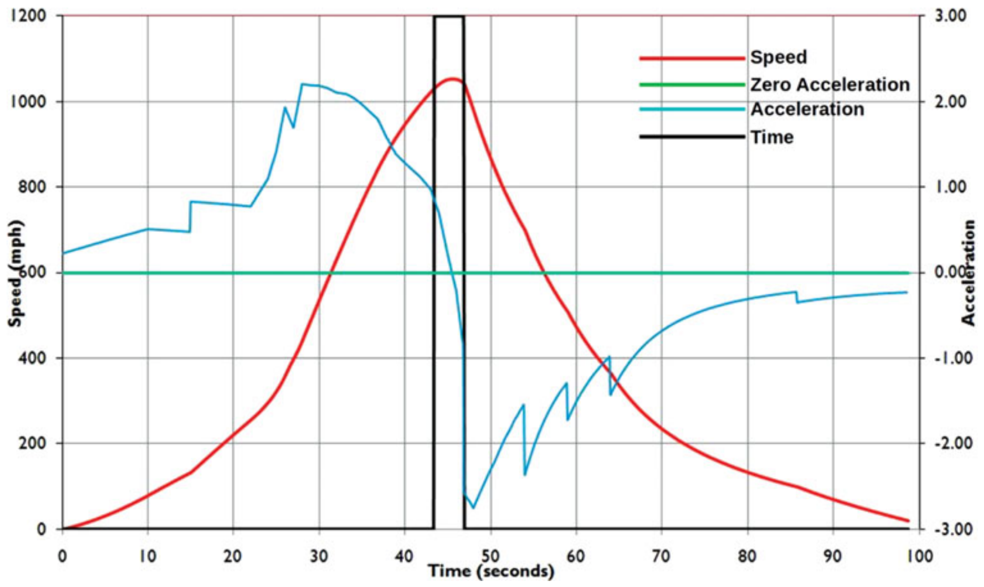

Figure 3. (Colour online) A 1000 mph run profile of BLOODHOUND SSC. Note that airbrake deployment commences during the deceleration phase at approximately 55 seconds and airbrakes are fully deployed by the time the vehicle has decelerated to 500 mph.

the vehicle down after reaching its peak speed. These airbrakes, situated on either side of the main body of the vehicle will be deployed at approximately 55 seconds into the run. The effect of airbrake deployment can clearly be seen during the deceleration phase of the run profile.

This application was deemed to be a perfect test case for the visualisation technique detailed in this paper due to the vehicle's geometrical complexity and, due to unique aerodynamic challenge of taking a land vehicle faster than the speed of sound, the designers' lack of prior knowledge of likely aerodynamic performance and anticipated aerodynamic performance. It was hoped that the approach developed for this project would significantly improve the aerodynamic designers' understanding of the phenomena that would determine the car's performance and, in turn, success in its record breaking attempt.

1.2 Flow visualisation of CFD data

The challenges of visualising CFD simulations include handling large, unstructured, high-dimensional data. Although many algorithms have been described for the placement of streamlines, relatively few have been presented for stream surfaces. In this paper, recent work on automating the placement of stream surfaces is adapted(7-9) and the effectiveness of this approach in the context of engineering design is demonstrated. Visualising CFD results by means of the rendering of stream surfaces has become increasingly popular in recent years. Stream surfaces have important inherent characteristics that can enhance the visual perception of complex flow structures(6,19,30,31). A stream surface is the integration through 3D steady flow of all points starting on a curve. The resulting surface is everywhere tangent to the local flow. They are useful for separating distinct regions of similar flow behaviour.

Industrial CFD applications, such as BLOODHOUND SSC, have specific challenges associated with them in terms of flow visualisation including high computational mesh resolution and high levels of adaptivity in the mesh. An efficient and robust system for interpreting the vector field resulting from the CFD analysis is therefore highly desirable. The requirements of design engineers making use of CFD simulations include visualising large datasets, user-guided semi-automatic seeding of surfaces to represent interesting subsets of the flow, and locating and visualising areas of turbulent flow.

The work presented in this paper is an adaptation of a previous surface placement algorithm(8) such that it is more applicable to large ($>$10,000,000 elements), unstructured CFD simulation data with emphasis on reducing the memory requirements and computational time for the clustering, while processing unstructured data efficiently. Part of the motivation behind our work is a request from industrial design engineers utilising CFD datasets to interactively select a sub-set (or cluster) of the CFD data and obtain more details.

In order to accelerate computational speed, and process the large datasets resulting from BLOODHOUND SSC CFD simulations, an algorithm is introduced based on K-Means clustering. K-Means clustering is able to partition the flow into subsets of data defined by a customised distance function. To produce meaningful subsets comparable to previous work(8), a derived curvature magnitude field, a velocity gradient magnitude field, and a Euclidean distance field are combined using a weighted linear distance function. It is the resultant scalar field which is clustered. The distance function uses weighting parameters to enable the user to guide the final visualisation results.

To aid the capture of the underlying flow structures, seeding curves derived from the curvature field are used. A seeding curve is placed at the centre of each cluster. This technique is used by Edmunds et al(8); however, here a modification is made utilising an adaptive step integrator to refine the curve in complex areas. This results in a smoother stream surface. The clustering technique is used to locate areas of interest for the design engineer and then to generate seeding curves and surfaces associated with a given cluster, which yield insightful representations of the CFD data. The effectiveness of this approach in the context of aerodynamic design is demonstrated by a presentation of its impact on the design of the airbrake doors of BLOODHOUND SSC.

1.3 Literature review

The related work falls into the following categories: streamline placement, stream surface placement and vector field clustering. The reader is referred to Edmunds et al(6) for a complete overview of flow visualisation with surfaces.

1.3.1 Streamline placement

Algorithms for the automatic seeding of streamlines have a long and rich history(5,17,21-23,40,42,43,48). Stream surface placement is the next logical phase in the evolution of flow visualisation.

Streamline placement does not translate directly to the placement of stream surfaces. Constructing a surface from discontinuous seed locations will produce inconsistent surfaces that may convey little meaningful information. To construct well-formed surfaces, the seeding structure should be smooth, continuous and represent the underlying flow.

1.3.2 Surface placement strategies

Theisel et al's approach to constructing saddle connectors in place of separating stream surfaces addresses the challenges of occlusion in 3D flow visualisation(38). This was extended by Weinkauf et al(44) using separating stream surfaces originating from boundary switch curves. A *boundary switch curve* represents the location where outflow at the boundary plane switches to inflow, and vice versa. One limitation of boundary switch curves is their possible absence in a given flow. Peikert et al(28) present topologically relevant methods for constructing stream surfaces which visualise critical points, periodic orbits, unbounded curvature and tightly winding spirals. The work by Edmunds et al(9) presents an automatic seeding algorithm for stream surfaces seeded at the boundary of the domain. Isolines are derived at the boundary based on exit flow trajectory. These isolines are then used to seed surfaces. The isolines are based on the idea of boundary switch curves of Weinkauf et al(44).

Edmunds et al(8) present an approach to surface placement utilising the hierarchical clustering technique described by Telea and Van Wijk(36). Seeding curves are computed based on the curvature of the flow field, from which surfaces are constructed and rendered. Reducing the memory footprint and increasing computational speed for use with larger unstructured CFD data is our motivation for extending this work. The three main constituents of the distance function in Edmunds et al(8) are Euclidean distance, vector direction and vector magnitude. As stated in Edmunds et al(8), the main idea is to guide the clusters to curved areas of flow where there is a change in geometric curvature of the flow (associated with vector direction) and a change in the velocity magnitude gradient. Based on a similar strategy a novel customised distance function combining flow curvature, velocity gradient and Euclidean distance is introduced.

1.3.3 Clustering

Xu and Wunsch(47) survey the topic of clustering, focusing on scalar clustering algorithms rooted in statistics, computer science and machine learning. Clustering algorithms can be divided into hierarchical clustering or partition-based clustering. The Telea and Van Wijk(36) algorithm utilises a hierarchical approach, as does the work by Edmunds et al(8). These algorithms are effective in providing a simplified representation of a vector field. However, these algorithms are $O(n^2)$ complex, where n is the number of initial samples. This has a significant impact on computation and memory requirements. Alternatively, partitioning algorithms such as K-means clustering(2) are generally $O(i \cdot k \cdot n)$ complex(18), where k is the number of centroids or means, and i is the number of iterations. K-means clustering is utilised to partition the domain using a novel distance function combining flow curvature magnitude, velocity gradient magnitude, and normalised Euclidean distance. K-means is suitable for large, unstructured CFD datasets due to its computational efficiency and small memory footprint.

1.3.4 Spatial hash grids

For sampling unstructured data a regular grid, Octree, BSP tree or other data structure storing pointers to intersecting tetrahedral cells may be used, reducing the search space when sampling the dataset. To improve memory usage and maintain good sampling speed spatial hash grids are utilised. Teschner et al(37) propose an approach for collision and self-collision detection of dynamically deforming objects that consist of tetrahedrons. The algorithm employs a hash function for compressing a regular spatial grid. The hash function can be generated very efficiently and does not require complex data structures, such as Octrees or BSPs. The authors investigate and optimise the parameters of the collision detection algorithm, such as the hash function, hash table size and spatial cell size. Following this work, Eitz et al(10) present a hierarchical spatial hash grid scheme for real-time collision detection. The authors employ an infinite hierarchical spatial hash grid in which for each single tetrahedron in the scene a well-fitting grid cell size is computed. A hash function is used to project occupied grid cells into a finite 1D hash table. This hierarchical spatial hash grid scheme is employed for the fast, memory-efficient sampling of an unstructured tetrahedral mesh.

In summary, the main benefits and contributions of this paper are:

- A tailored algorithm, developed specifically to support design decisions in aerodynamic design applications, which captures interesting subsets of the flow, while producing comparative results with previous hierarchical algorithms.
- Improved computational speed and memory usage over previous stream surface work, with the ability to process large unstructured datasets fast and efficiently.
- An algorithm to partition the flow field using K-means clustering, providing superior performance with less memory overhead, and a feature-based overview of the data.
- The novel use of a curvature magnitude field and a velocity gradient magnitude field combined with Euclidean distance for the clustering distance function.
- The application of this visualisation and interaction techniques to real world CFD data with a clear indication of the impact this has had on engineering design.

The rest of this paper is divided into the following sections. The remainder of this section is dedicated to a review of the literature related to flow visualisation to provide a context for the visualisation algorithm presented. A review of the CFD process is given in Section 2. A detailed presentation of the algorithm is given in Section 3. The results are reviewed in Section 4 and the impact of this visualisation approach in the context of the design of BLOODHOUND SSC is provided in Section 5. Conclusions and future work are discussed in Section 6.

2.0 THE CFD PROCESS

CFD simulations of the BLOODHOUND vehicle were performed using the in-house FLITE3D system(12). The FLITE3D solver is a vertex-based, finite-volume solver that uses agglomeration-based multi-grid algorithms on unstructured hybrid meshes. The system is based on solution of the steady-state Reynolds-averaged compressible Navier-Stokes equations with a range of turbulence model options including Spalart-Allmaras(33), K-Omega(46) and Menter-SST(24). Note that the use of turbulence model in the CFD simulation has no implications on the visualisation strategy detailed in this paper.

Stabilisation of the solver for high-Mach-number simulations is achieved by replacing the inviscid finite-volume fluxes with a consistent second-order HLLC flux function. Local

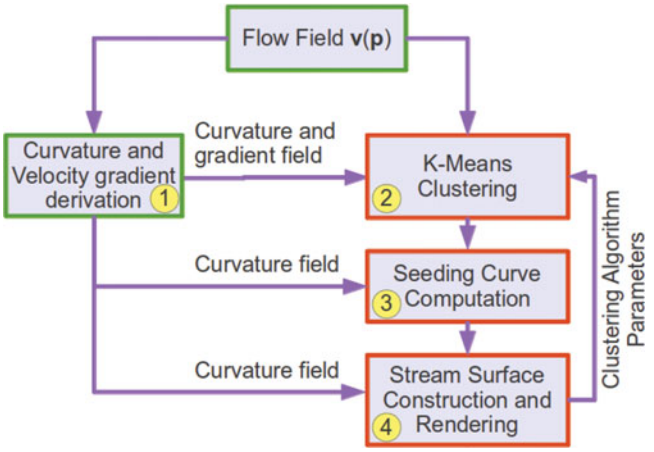
Flow Field v(p)
Curvature and Velocity gradient derivation 1
Curvature and gradient field
K-Means Clustering 2
Curvature field
Seeding Curve Computation 3
Curvature field
Stream Surface Construction and Rendering 4
Clustering Algorithm Parameters

Figure 4. (Colour online) The automated stream surface seeding pipeline. The pipeline shows the curvature field and velocity gradient field derived from the flow field. These are used as inputs to the clustering, seeding curve generation and illustration techniques. Stream surfaces are propagated from the seeding curves through the vector field and then rendered.

time integration to steady state is implemented using a three-step Runge-Kutta algorithm. Appropriate boundary conditions were applied, and these are detailed in Evans et al(12). Simulations were typically run on meshes of between 10 m and 100 m finitevolume cells on 128 cores of a PC cluster. Run times to a three orders of magnitude density residual convergence typically would take in the region of a 24 to 48 hours wall clock. The level of mesh resolution to ensure mesh convergence for a given simulation was pre-determined in a separate study. Details of this and a more in-depth discussion of the CFD approach can be found in Evans et al(12).

3.0 METHODOLOGY – STREAM SURFACE PLACEMENT

This section describes our adapted stream surface seeding algorithm, starting with an overview of the pipeline illustrated in Fig. 4. The algorithm features K-means clustering of combined Euclidean distance, curvature magnitude and velocity gradient magnitude. This is in contrast to Ref. 8, which hierarchically clusters combined velocity direction and magnitude with Euclidean distance direction and magnitude. A new distance function and associated weightings is used to combine the data attributes, and thus guide the results. Seeding curve generation starts from the cluster centres as in Ref. 8. However, the discretised seeding curve construction is further refined to better represent more complex areas of the velocity field.

The input to our visualisation framework is an unstructured tetrahedral meshed CFD simulation output file(12). The input to the algorithm is $\mathbf{v}(\mathbf{p}) \in \mathbb{R}^3$, where $\mathbf{v} \in \mathbb{R}^3$, $\mathbf{p} \in \Omega$ and is an unstructured tetrahedral mesh in $\mathbb{R}^3$.

1. A curvature field and a velocity gradient field is first derived. These fields are derived directly from the velocity field. The curvature field is also used to compute seeding curves after the clustering process has identified the seeding locations and is used to map opacity to stream surfaces for illustrative rendering. These derived fields are saved to disk for quick loading by the user on subsequent analysis. See Section 3.1.

2. The domain is then partitioned into k clusters where k is user defined. The data array indices of the vertices representing the scalar attributes are stored. The K-means algorithm iterates until it converges to a stable set of means. Flow curvature, velocity gradient and Euclidean distance are comparison attributes that can be set. See Section 3.2.
3. The seeding curves are then computed at the cluster centres. The cluster centre is automatically used as the basis of the seeding curve. The seeding curves are then generated by integrating forward and backward through the curvature field at a length proportional to the cluster size. This generates seeding curves which follow the local flow structures while maintaining orthogonality with the flow. See Section 3.3.
4. Stream surfaces are propagated from each of the seeding curves. Flow attributes may be mapped to colour and opacity. After the generation of the stream surfaces, the surface data is rendered using a number of illustrative techniques to enhance perception. See Section 3.4.

This pipeline differs from Ref. 8 in that the derived fields are computed first as input to the clustering process, the derived fields are used in combination with the new distance function to generate clusters in areas where there is a non-zero velocity gradient, and a concentration of clustering in areas of high curvature.

Following this, the data is clustered with the K-means clustering algorithm rather than the hierarchical clustering proposed in Ref 8. This reduces the computational and memory requirements, and enables the clustering of unstructured CFD data without modification. The seeding curves are then generated in a similar fashion to Ref. 8, modified to use an adaptive integration method. The motivation for this is that it increases the accuracy of the initial surface representation in complex areas.

Finally, the stream surfaces are constructed and rendered using the same illustration techniques as in the previous work in Ref 8. These methods are effective for capturing the characteristics of the flow when applied to stream surfaces.

3.1 Derived fields

From the vector field $\mathbf{v}(\mathbf{p})$, a curvature field $\mathbf{c}(\mathbf{p})$ and a velocity gradient field $\mathbf{g}(\mathbf{p})$ is derived. The role of $\mathbf{c}(\mathbf{p})$ and $\mathbf{g}(\mathbf{p})$ is to support the clustering process (Section 3.2). $\mathbf{c}(\mathbf{p})$ also supports the seeding curve computation (Section 3.3) and illustrative techniques for rendering the stream surfaces (Section 3.4).

The curvature field $\mathbf{c}(\mathbf{p}) \in \mathbb{R}^3$, where $\mathbf{c} \in \mathbb{R}^3$, is derived by applying a combination of operators to the vector field. $\mathbf{c}(\mathbf{p})$ is derived from $\mathbf{v}$ and the second derivative $\mathbf{a}$ of the flow field. The second derivative $\mathbf{a}$, acceleration, is defined as $\mathbf{a} = (\nabla\mathbf{v})\mathbf{v}$, where $\nabla\mathbf{v}$ is the Jacobian of the velocity field. Steady-state curvature(32) is defined as $|\mathbf{c}|$, where:

$$\mathbf{c} = \frac{\mathbf{v} \times \mathbf{a}}{|\mathbf{v}|^3} \qquad \ldots(1)$$

The velocity gradient field $\mathbf{g}(\mathbf{p})$, where $\mathbf{g} \in \mathbb{R}^3$, is derived from the velocity field as follows:

$$\nabla|\mathbf{v}| = \frac{\partial|\mathbf{v}|}{\partial x}i + \frac{\partial|\mathbf{v}|}{\partial y}j + \frac{\partial|\mathbf{v}|}{\partial z}k, \qquad \ldots(2)$$

where $\mathbf{g} = \nabla|\mathbf{v}|$, i.e. the gradient of the velocity magnitude, and i, j, k are the components of a unit vector. The gradient field is computed to support the clustering process (Section 3.2).

3.2 K-means clustering

Vector field clustering has been used to show interesting flow features for real-world datasets(29). It offers the benefit of not having to make a binary decision based on the presence of a feature. K-means clustering algorithms provide a general approach to partitioning data. With relatively low computational requirements and memory usage compared to hierarchical clustering, K-means is a good candidate to solve data partitioning. This algorithm needs no modification to deal with either structured or unstructured data. This approach enables the aerodynamic designer to focus on an individual structure in the flow.

K-means is one of the simplest unsupervised learning algorithms that solves the well-known clustering problem(2). The procedure provides a way to classify a given data field into k clusters chosen a priori by the user. The idea is to define k centroids, one centroid c for each cluster (or partition) π. The reader is referred to Section 4 for a discussion of the choice of k. The next step is to take each grid vertex $\mathbf{p}_i$ belonging to Ω and associate it with the nearest initial centroid. The procedure iterates until it converges to a stable set of centroids (or means) and partitions. It is important to note that the clustering is performed in scalar space utilising the customised distance function described next.

3.2.1 *A customised distance function*

Edmunds et al(8) guides the clustering process to areas of flow where there is a change in geometric curvature e.g. difference in velocity vector direction (η_δ), and velocity gradient (η_μ). These attributes combined with position (η_ψ) form the basis of the distance function. The distance function can be altered by modifying the bias between each of the three components.

Their aim is to generate seeding curves in optimal locations adjacent to the flow structures, with minimal seeding density. Interesting flow structures may be characterised by their curvature and velocity gradients. Another aim is to generate clusters in areas of non zero velocity gradient and a concentration of clustering in areas of high curvature.

For this specific application a novel distance function $d(c,\ s)$ is introduced combining flow curvature magnitude $|\mathbf{c}|$, velocity gradient magnitude $|\mathbf{g}|$ and Euclidean distance magnitude $|\mathbf{e}|$, where $\mathbf{e} \in \mathbb{R}^3$, for use with K-Means clustering.

To incorporate the curvature of the flow, velocity gradient, and Euclidean distance into a single unified distance function their relationship is specified as a simple linear combination:

$$l(|\mathbf{c}|, |\mathbf{g}|, |\mathbf{e}|) = |\mathbf{c}| + |\mathbf{g}| + |\mathbf{e}| \qquad \ldots (3)$$

This combination alone does not enable flexibility for fine tuning the results in marginal cases where either flow curvature or velocity gradient strongly influences the results. Additionally the ability to produce both focus and contextual visualisations is desired for the domain user. This motivated the addition of user-specified bias to the distance function. To control the influence of $|\mathbf{c}|$ or $|\mathbf{g}|$, a linear relationship is specified:

$$i(|\hat{\mathbf{c}}|, |\hat{\mathbf{g}}|) = B|\hat{\mathbf{c}}| + (1 - B)|\hat{\mathbf{g}}|, \qquad \ldots (4)$$

where $B \in [0, 1]$ and $\hat{\mathbf{c}}$ and $\hat{\mathbf{g}}$ are range normalised e.g. $|\hat{\mathbf{c}}| \in [0, 1]$ and $|\hat{\mathbf{g}}| \in [0, 1]$. To influence a focus or contextual bias, a linear relationship is specified between i and $|\hat{\mathbf{e}}|$, where $A \in [0, 1]$

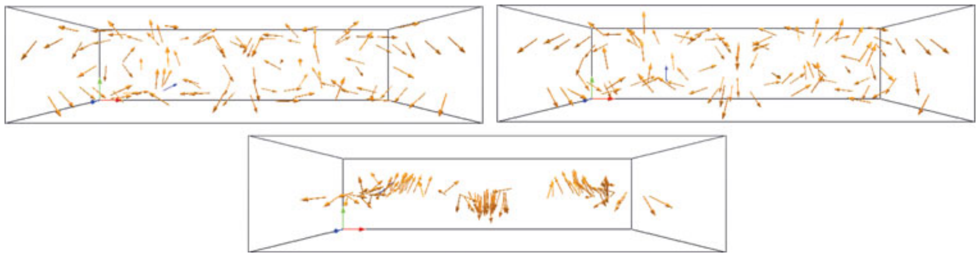

Figure 5. (Colour online) This image shows the Bernard flow simulation clustered with our algorithm. The cluster centres are represented by the base of the arrow glyphs where each glyph shows vector direction at that location. All images are clustered with $B = 0.5$, and $k = 100$. The top left image is clustered with parameter $A = 0.05$, and shows a more evenly distributed set of clusters as Euclidean distance is emphasised. The top right image is clustered with $A = 0.5$, and the bottom image is clustered with $A = 0.95$ showing the emphasis moving towards the centre of the curved flow with higher values of A.

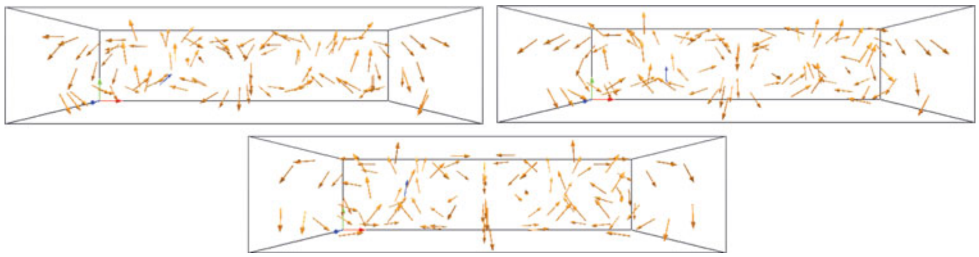

Figure 6. (Colour online) All images are clustered $A = 0.5$, and $k = 100$. The top left image is clustered with parameter $B = 0.1$. The top right image is clustered with $B = 0.5$, and the bottom image is clustered with $B = 0.9$. The changes are more subtle when emphasising velocity gradient over flow curvature.

and $\hat{\mathbf{e}}$ is normalised by the domain extents e.g. $|\hat{\mathbf{e}}| \in [0, 1]$:

$$l(|\hat{c}|, |\hat{g}|, |\hat{e}|) = Ai + (1 - A)|\hat{e}| \qquad \ldots (5)$$

The distance function $d(c, s) = ||c - s||^2$ determines how the domain is partitioned. Both the centroid $c(\pi)$ and the attribute $s(\mathbf{p})$ are scalar values computed from the linear relationship $l(|\hat{\mathbf{c}}|(\mathbf{p}), |\hat{\mathbf{g}}|(\mathbf{p}), |\hat{\mathbf{e}}|(\mathbf{p}))$. By changing the parameters A and B, different insights into the characteristics of the flow behaviour can be communicated. For example, increasing the value of A will provide a more focused visualisation. Conversely, reducing A will provide increased context to the visualisation. Adjusting B will adjust the influence either $|\mathbf{c}|$ or $|\mathbf{g}|$ has towards the final visualisation.

3.2.2 User input parameters

The choice of user input parameters A, B, and k will affect the resulting visualisation. In this section, the effect that the different parameters have on the distribution of the clusters is shown. It can be seen in Fig. 5 that as the parameter A is decreased it provides a more evenly distributed set of clusters which provide a good overview of the characteristics of the flow field. In contrast, it can be seen that an increase of A results in a more focused distribution of clusters in areas of high curvature. The change in emphasis between flow curvature and velocity gradient enables the user to fine tune the clustering. The effects of changing parameter B is demonstrated in Fig. 6.

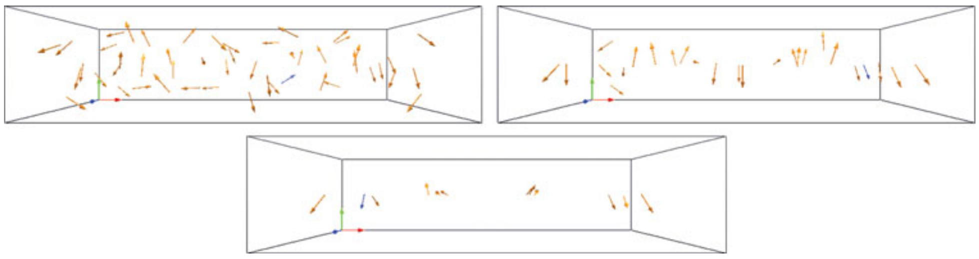

Figure 7. (Colour online) All images are clustered $A = 0.5$, and $B = 0.5$. The top left image is clustered with a k of 50, the top right with a k of 25, and the bottom image with a k of 12. As the quantity of clusters reduce a more focused distribution is observed.

Another important parameter is the choice of k. Too many clusters will produce a cluttered visualisation, and too few will not adequately capture the interesting features within the flow field. The effect of changing k is shown in Fig. 7. Choosing the optimum number of clusters for a given case is a difficult problem. The same is true for the general case of K-means clustering. There is much literature on this matter, as discussed in Fang et al[13] and Ferreira et al[14]; however, the authors are not aware of any definitive approach to solving this problem. The visualisations are therefore demonstrated, highlighting the number of chosen k, to provide guidance to the aerodynamic designer. Parameter k is a user option.

3.2.3 K-means clustering algorithm

For a set of scalars $\mathcal{S} = \{s(\mathbf{p}_1), \ldots, s(\mathbf{p}_n)\} \subset \mathbb{R}$, a distance function $d(c,\ s(\mathbf{p}_i))$ defines a centroid $c = c(\mathcal{S}) \subset \mathbb{R}$ of the set $\mathcal{S}$ as a solution of the minimisation problem:[18]

$$c = arg\,min \left\{ \sum_{s(\mathbf{p}_i)\in\mathcal{S}} d(c,\ s(\mathbf{p}_i)) \right\} \qquad \ldots (6)$$

Let $\prod = \{\pi_1, \ldots, \pi_k\}$ be a partition of $\mathcal{S}$, that is:

$$\bigcup_i \pi_i = \mathcal{S}, \quad \text{and} \quad \pi_i \cap \pi_j = 0 \quad \text{if} \quad i \neq j \qquad \ldots (7)$$

Given a partition $\prod = \{\pi_1, \ldots, \pi_k\}$ of the set $\mathcal{S}$, one can define the corresponding centroids $\{c(\pi_1), \ldots, c(\pi_k)\}$ by:

$$c(\pi_i) = arg\,min \left\{ \sum_{s(\mathbf{p}_i)\in\pi_i} d(c,\ s(\mathbf{p}_i)) \right\} \qquad \ldots (8)$$

For a set of k centroids $\{c_1, \ldots, c_k\}$, one can define a partition $\prod = \{\pi_1, \ldots, \pi_k\}$ of the set $\mathcal{S}$ by:

$$\begin{aligned} \pi_i = \{s : s \in \mathcal{S},\ d(c_i,\ s) \leq d(c_j,\ s) \\ \text{for each } j = 1, \ldots, k\}, \end{aligned} \qquad \ldots (9)$$

noting that in general $c(\pi_i) \neq c_i$.

3.3 Seeding curve computation

The seeding curve generation commences following the clustering. The final set of k centroids $\{c_1, \ldots, c_k\}$ are used to define the location for the origin of each seeding curve. Seeding curves are generated using the same method as in Ref. 8. The seeding curves are generated by integrating through the curvature field. Length is restricted to the cubic root of the cluster volume:

$$length(\pi) = \sqrt[3]{vol(\pi)}, \qquad \ldots (10)$$

where $vol(\pi)$ is the spacial extent of the cluster (or partition) $\pi(c_k)$. The integration uses a fourth-order Runge Kutta integrator, sampling the curvature field, in the forward and backward directions. The original approach is modified by using an adaptive fourth-order Runge Kutta integrator which provides a discretised seeding curve with a denser set of vertices in areas of increased complexity. The motivation for this is that it increases the accuracy of the surface representation in complex areas.

3.4 Surface construction and rendering

For the sampling of unstructured tetrahedral data, a hierarchical spacial hash grid is used as described in Ref. 10. This method is employed for fast look-up of vertex to tetrahedral cell intersection and interpolation. For each single tetrahedron in the domain, a well-fitting grid cell size is computed. A hash function is used to store a reference to the occupied grid cells in a finite 1D hash table. The hash is computed from the bounding box of the tetrahedral cell. When performing look-up, the hash is computed from the sample vertex. This method reduces the memory footprint for cell storage, while maintaining fast look-up for intersection and inter-polation tests. In the case of the full BLOODHOUND SSC CFD data, the resolution of an equivalent regular grid fine enough to capture a similar quantity of tetrahedral cells per grid location, would be 13,607 × 13,607 × 6,792. At one 64-bit pointer per grid cell, an approximate memory usage would be 9,000 GB, not including any other storage overhead. However, the hierarchical spacial hash grid, at one pointer and one integer per tetrahedral cell, uses approximately 500 MB not including any other storage overhead. The storage overhead for the hierarchical spacial hash grid relies on a hash map/binary tree implementation for fast look-up. This is more expensive than a simple array for the regular grid. Using standard C++ STL map containers, the memory overhead rises to approximately 2.5 GB.

Our work utilises an existing solution for generating stream surfaces(15). An adaptive fourth-order Runge Kutta integrator is used in the surface construction. The user can select downstream and upstream propagation. Surfaces are terminated when they exit the domain, enter a periodic orbit, or reach a pre-determined maximum length. The user has an option to control the length.

A number of techniques are implemented to aid the viewer in perception of the resulting visualisation. Options include the use of transparency, colour and silhouette edge highlighting. Transparency in visualisations pose problems relating to the order of primitive rendering. Depth peeling, an order-independent transparency technique presented by Bavoli and Myers(1), is used. Silhouette edge highlighting is used to help the viewer in perceiving where the surfaces curve away from the viewer and to enhance surface edges. Silhouette highlighting utilises a Gaussian kernel in image space(25). Reducing the saturation of colour as the surfaces curve away from the viewer further enhances the perception of shape(8).

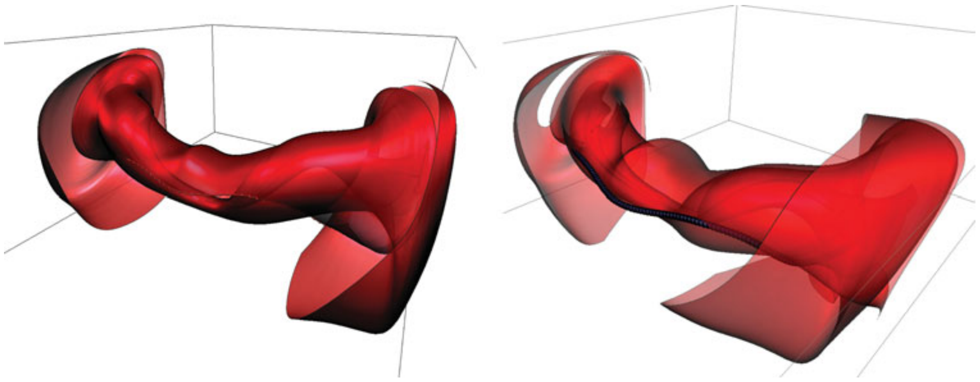

Figure 8. (Colour online) The left illustration of the Bernard simulation demonstrates the seeding curve following the flow structure. The seeding location is derived from the clustering with parameters $A = 0.5$, $B = 0.5$ and $k = 4$. Three of the surfaces are hidden in the rendering. The right illustration of the Bernard simulation is provided for comparison courtesy of Edmunds et al[8].

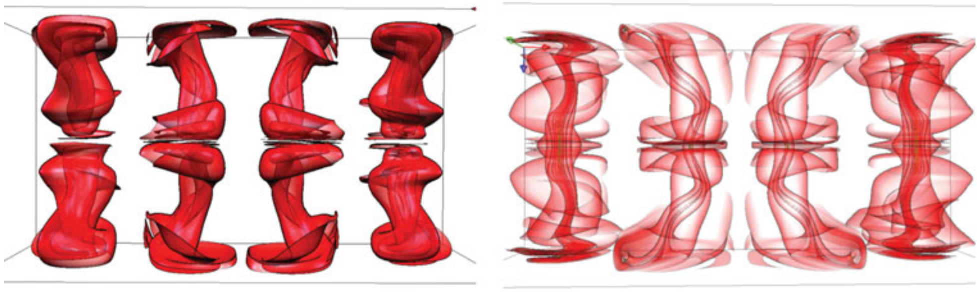

Figure 9. (Colour online) The left illustration is the Bernard flow numerical simulation visualised using our seeding algorithm. The seeding locations are derived from the clustering process using parameters $A =$ 0.4, $B = 0.5$ and $k = 8$. The four main double vortex structures are clearly emphasised by the eight surfaces. The thermal motion of the flow field is captured with our framework. The figure shows the surfaces rendered with transparency. The right illustration of the Bernard simulation is provided for comparison courtesy of Edmunds et al[8].

4.0 RESULTS

Here we evaluate the algorithm based on its predecessor and application to the BLOODHOUND SSC.

4.1 Visual comparisons

Before application to the complex geometric example of BLOODHOUND SSC, in this section the algorithm is compared to the previous work of Edmunds et al[8]. The ability to capture the same features found within each of the datasets is demonstrated. First the Bernard flow numerical simulation defined on a regular grid[45] is presented. The simulation demonstrates thermal motion as a result of convection. Our algorithm captures the double vortex structures using parameters $A = 0.5$, $B = 0.5$ and $k = 4$. As shown in Fig. 8, one of the double vortex structures, which compares well with Fig. 9 from Ref. 8, is illustrated. A comparative visualisation of the full Bernard illustration as shown in Fig. 9 is also presented. This visualisation is generated from parameters $A = 0.4$, $B = 0.5$ and $k = 8$. This visualisation compares well with Fig. 15 from Ref. 8 and has the advantage of seeding each side of the double vortices separately. This is useful for a more detailed exploration of the flow

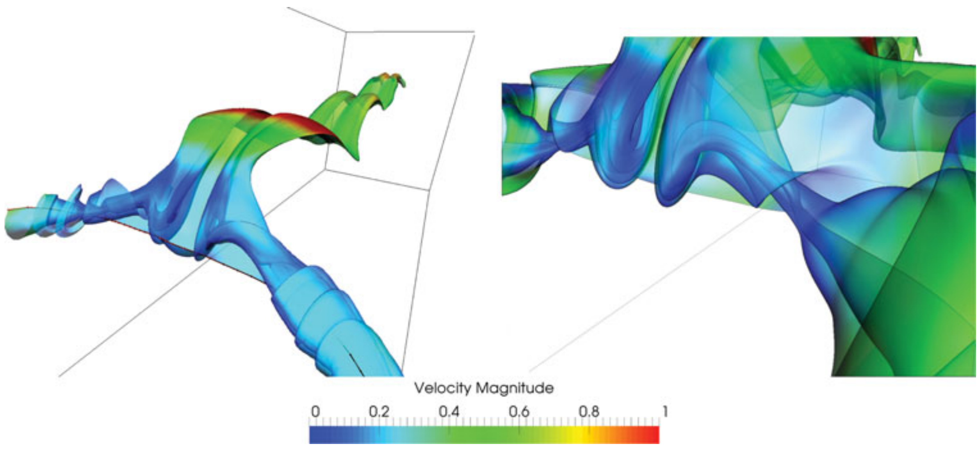

Figure 10. (Colour online) The dataset shown here is a direct numerical Navier-Stokes simulation by Simone Camarri and Maria Vittoria Salvetti (University of Pisa), Marcelo Buffoni (Politecnico of Torino), and Angelo Iollo (University of Bordeaux I)(4), which is publicly available(39). A uniformly re-sampled version is used, which has been provided by Tino Weinkauf and used in von Funck et al for smoke surface visualisations(41). The left illustration of flow past a cuboid simulation demonstrating a stream surface generated near a double vortex structure emanating from a critical point. The clustering parameters used for this visualisation are $A = 0.9$, $B = 1.0$ and $k = 3$. The right illustration of the cuboid simulation is provided for comparison courtesy of Edmunds et al(8). Colour is mapped to range normalised velocity magnitude.

structures. The flow past a cuboid simulation as in Fig. 10 is presented next. This visualisation demonstrates the ability of our algorithm to capture the double vortex structure downstream from the cuboid. In summary, our algorithm compares well with the approach of Ref. 8. The same underlying characteristics of the flow fields are captured with improved computational speed while maintaining a smaller memory footprint.

4.2 Memory and performance evaluation

The proposed method is compared with the work in Ref. 8. The hierarchical clustering is $O(n^2)$, compared with our algorithm whose complexity is $O(i \cdot k \cdot n)$, where n is the number of initial samples. Our algorithm is implemented in C++ and QT4 on a PC with an NVIDIA GeForce GTX480, an Intel quad core 2.8 GHz CPU and 8 GB RAM. The data is loaded using either the freely available TecplotIO library(35) or by loading the Swansea University FLITE(34) CFD simulations directly. The bottleneck in the performance of our algorithm is the clustering as seen in Table 1. The performance of stream surface rendering is comparable to previous work e.g. Born et al(3), Hummel et al(16) and Edmunds et al(8).

The memory requirement for our approach is proportional to the size of the dataset e.g. n, where n is the number of initial samples. Only one integer per vertex is stored to reference the cluster it belongs to. The hierarchical approach stores vector data, location data, cluster size data, cumulative error data and neighbourhood/connectivity information at each node in the binary tree. The quantity of nodes is $2n - 1$. The K-mean clustering with the customised distance function approach can perform up to one order of magnitude faster than its predecessor.

Table 1
Clustering performance of a range of simulations

	Clustering Performance			
Data	**Elements**	***k***	**Time(ms)**	**[8](ms)**
Bernard	$128 \times 32 \times 64$	4	1246	2908
Cuboid	$192 \times 64 \times 48$	3	2095	6898
Lorenz	128^3	5	11,058	34,812
Airbrake	5,764,071	10	10,795	n/a
BLOODHOUND	40,963,951	5	37,011	n/a

All are regular grid data except the BLOODHOUND SSC data, which is an unstructured tetrahedral grid. For reference, the last column contains comparative times from Ref. 8.

5.0 APPLICATION AND IMPACT: BLOODHOUND SSC

In the following section, the impact of the visualisation algorithm is demonstrated in the context of the aerodynamic design of BLOODHOUND SSC using CFD simulation as the primary tool to guide the design evolution.

Ultimately the behaviour of aerodynamic bodies such as BLOODHOUND SSC is governed by the force coefficients that they generate at a given flight condition e.g. lift coefficient as a function of Reynolds number, Mach number and angle-of-attack, or drag coefficient as a function of Mach number and airbrake deployment position. However, it is often very difficult to determine why a force coefficient varies from one design configuration to another using the traditional flow visualisation methods of pressure coefficient plots over body surfaces or cuts through the domain, or glyphs at the computational sampling points (for reasons already mentioned). An ability to relate the variation of force coefficients (and related quantities of interest such as ‘centres of pressure’) to the underlying causal flow phenomena such as flow separation, vortex shedding and shock wave formation is important in making informed design decisions. Traditionally, determining the location for seeding of streamlines and surfaces presents a significant challenge both in terms of the time required by the aerodynamic designer to identify the important flow features and also in terms of consistency when comparing simulation data for different engineering designs (or configurations) that have been computed on different meshes. Since aerodynamic phenomena that can affect the macroscopic behaviour of the body often originate at the very small scale e.g. flow separation/detachment or vortex shedding, the detection (or non-detection) of such features is often highly sensitive to the initial seeding of streamlines and stream surfaces. The approach presented in this application provides a way towards eliminating both of these issues: speed and consistency in analysis. Note that there is a great benefit that the user still has the ability to interact with the visualisation algorithm via the parameters A, B and k and maximum surface length in order to tune the resulting visualisation, but holding these parameters constant whilst comparing two designs provides a level of consistency often not available when the user is manually determining the seeding positions. The automatic seeding of the stream surfaces from cluster centres ensures that for a given vector field dataset, there is the maximum probability of capturing the most important/significant features of the flow.

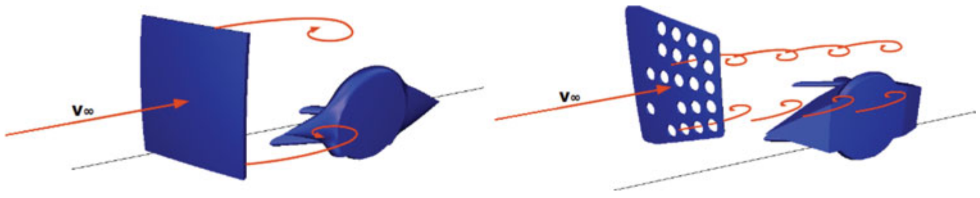

Figure 11. (Colour online) BLOODHOUND SSC CFD simulation geometry. The left image shows the initial airbrake design configuration with a solid construction, situated just in front of the rear suspension. The right image shows the final airbrake design configuration with holes, again situated just in front of the rear suspension.

Section 3.2, paragraph "User Input Parameters" discusses the effect of changing the parameters with regards to guiding the algorithm for a focus or context examination of the results. Higher values for A provide a more focused result i.e. concentrates on areas of higher curvature and velocity gradient. Conversely, lower values tend not to focus and produces results which are more evenly distributed, providing the user with better contextual information. Higher values for the B parameter guides the focused results towards curved aspects of the flow field, while lower values move the results towards areas of higher-velocity gradients. With regards the k value, higher numbers simply provide more centroids and thus more seeding locations. As a rule of thumb, values of $A = 0.5$, $B = 0.5$ and $k = 10$ generally provide adequate results as a starting point for unexplored simulation data. In the case of the airbrakes, where it is expected to see turbulent fluctuations and vortex shedding, higher values for A can be used to focus on the vortices and increasing the number of k will help capture more of the vortex structures.

The BLOODHOUND SSC case studies used to test the approach detailed here provides excellent tests for the usefulness of the approach. The first case considers the subsonic behaviour of the fully extended airbrake and how post-processing of CFD simulation data resulted in changes to the airbrake design. Two airbrake designs were considered, one being a solid 'door', and the second a door with holes/perforations inserted (see Fig. 11). One of the concerns about the aerodynamic behaviour of the airbrake system was regarding the characteristics of the downstream wake that would be generated. In particular, there were concerns about how this wake would interact with the complex rear wheel and suspension system downstream. It was important to ensure that the dominant vortex shedding frequencies from the airbrake were well above the natural frequency of the rear wheel system (approximately 10 Hz). This requirement led to the airbrake design with holes, which was an attempt to reduce the size and increase the frequency of the shed vortices.

The flow visualisation approach set out in this paper has been a significant step towards a better understanding of the complex underlying flow behaviour in the wake downstream of the airbrake. It has helped refine the position of the holes in the airbrake door to optimise the behaviour of the airbrake and minimise its impact downstream. In contrast, in the second case of supersonic flow over the full BLOODHOUND SSC vehicle with the airbrakes retracted, it is anticipated that flow separation and vortex shedding should be at an absolute minimum since this results in the minimum drag solution.

5.1 Airbrake design

Visualisation of the interaction of the vortices shed at separation positions on the airbrake with the downstream rear wheel and suspension identifies the power of the approach to represent

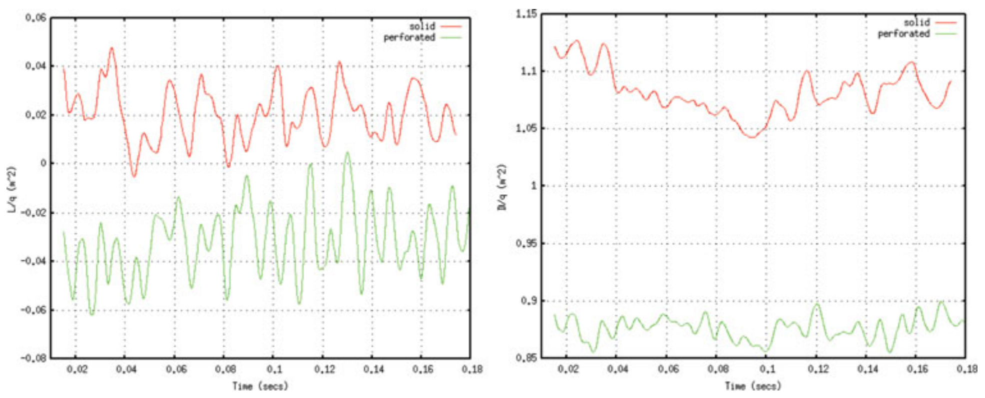

Figure 12. (Colour online) Transient force response of the solid and perforated airbrake designs. Lift and drag forces are shown normalised by free stream dynamic pressure, *q*.

this complex and unsteady flow field. It provides excellent guidance to the engineer on the implications of this design on its aerodynamic response.

Overall, the benefits of the approach proposed here are a speed-up of the visualisation component of post-processing CFD data for large datasets encountered in industrial CFD applications and thus a speed-up of the overall engineering design cycle, consistency when comparing engineering designs using flow visualisation, and the automatic detection of otherwise undetectable (or difficult to detect) flow phenomena.

The CFD data representing the BLOODHOUND SSC deployed airbrakes is a large unstructured tetrahedral mesh consisting of 5,764,071 elements. The simulation results include velocity ($\mathbf{v}_x$, $\mathbf{v}_y$, $\mathbf{v}_z$), density (ρ), coefficient of pressure (c_p) and energy (E). Please note all data values are range normalised due to confidentiality agreements. The memory footprint of the simulation data (post clustering) is approximately 0.7 GB. The clustering process takes 11 seconds. In this case study, results from two simulations is considered. The first is the initial design configuration of the airbrake, which is a solid ‘door’ situated upstream of the rear suspension assembly, and the second is a revised airbrake design which includes a set of holes. The CFD simulations revealed that the first airbrake design shed large, high-intensity vortex structures at a frequency close to the natural frequency of the rear suspension assembly.

A revised design was developed and optimised to include a set of holes which were predicted to produce a much higher frequency of smaller vortex structures greatly reducing the risk of exciting the natural frequency mode of the rear suspension assembly. See Fig. 11 for an image of the geometry used for these CFD simulation. This figure also demonstrates a simplified overview of aerodynamic phenomena revealed by the flow visualisation. The force coefficient response for these two designs is shown in Fig. 12.

Stream surface flow visualisation was used to gain an understanding of the flow physics generating the response in Fig. 12. This figure shows that the effect of adding the holes to the airbrake design is to increase the dominant vortex shedding frequency from $\approx$50 Hz to $\approx$150Hz i.e. further away from the natural frequency of the rear wheel structure downstream and thus reducing the likelihood of excitation. The cost of this is that the drag generated by the airbrake has been reduced by approximately 10%, but this drag loss was minimised by optimisation of the hole sizing and distribution across the airbrake face.

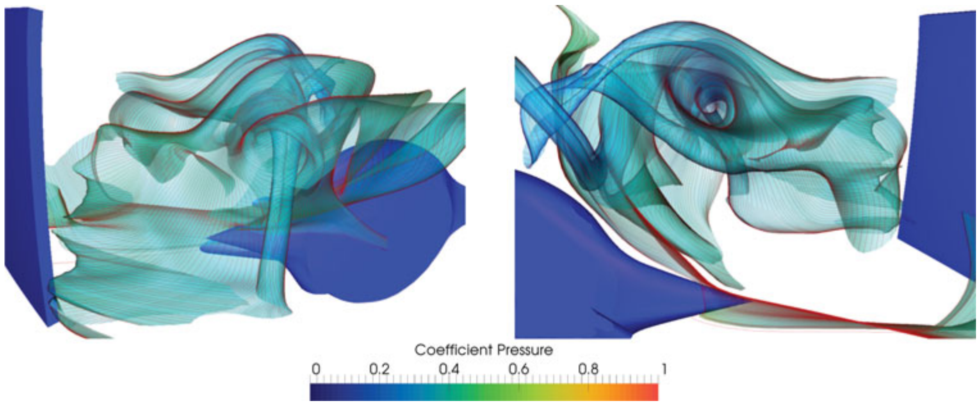

Figure 13. (Colour online) The left visualisation shows a close up of the solid airbrake. The large vortex structure can clearly be observed at the centre of this image. The right visualisation views the same vortex structure from the vehicle body. The throat of the vortex is located above the centre. Two further vortex structures can be seen – one formed around the top of the throat, and one formed around the bottom. Colour is mapped to range normalised c_p.

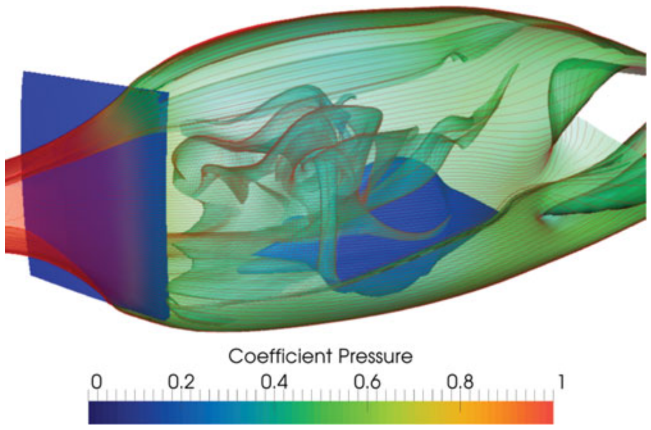

Figure 14. (Colour online) BLOODHOUND SSC CFD simulation of the the initial airbrake design configuration in use, clustered using parameters $A = 0.6$, $B = 0.6$ and $k = 13$. The image is using transparency to aid the visual perception of the flow behind the airbrake. A large vortex structure can be clearly seen left of centre. Colour is mapped to range normalised coefficient of pressure c_p, where free stream c_p is green, high c_p is red, and low c_p is blue.

For the first design simulation visualisation, a set of parameters was chosen which focuses on areas of high curvature with the aim of capturing large, high-intensity vortex structures behind the airbrake mechanism. The clustering results from parameters $A = 0.6$, $B = 0.6$ and $k = 13$ are illustrated in Figs 13 and 14. These visualisations confirm the presence of a large, high-intensity vortex structure travelling towards the rear suspension assembly. There is only one instance of a vortex structure in the space between the airbrake and past the rear suspension, indicating a lower frequency of vortex shedding. This visualisation of the CFD simulation clearly confirmed the design engineers' initial predictions regarding the cause of the response in Fig. 12.

For the second design simulation visualisation, parameters were adjusted to focus on the higher density of inter-woven curved flow structures forming behind the airbrake mechanism.

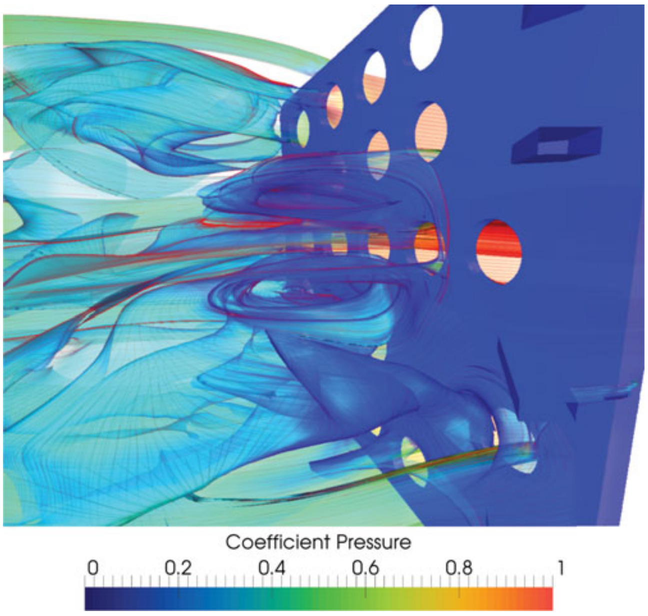

Figure 15. (Colour online) The visualisation shows a close up of the perforated airbrake viewed from the vehicle body. Two small vortex structures can clearly be seen at the centre of this image, forming just behind the airbrake. Colour is mapped to range normalised c_p.

The number of k to was increased achieve this, and the parameter A increased to further concentrate the clustering in curved areas as a reduction of extreme curvature found on the first simulation was expected. The clustering results from parameters $A = 0.95$, $B = 0.6$ and $k = 19$ are illustrated in Figs 15 and 16.

As expected, this is a result of the hole configuration causing the formation of small vortices being shed at high frequency from each of the holes. The high frequency of these low-intensity vortices quickly becomes inter-woven into an intricate area of turbulent flow. This flow pattern again confirmed the engineers' predicted behaviour and provided great insight into how the turbulent flow is forming and interacting with the suspension assembly and bodywork. This allowed the positioning and sizing of the holes on the airbrake door to be optimised to maximise drag whilst minimising the disturbance to the rear suspension structure.

5.2 Full vehicle aerodynamic performance

The aerodynamic objective of the full vehicle in its acceleration phase is very different to that of the airbrakes (which operate during the deceleration phase). Drag minimisation has been one of the main aerodynamic objectives for the full vehicle body. Minimising flow separation has therefore been highly desirable. As the design of the full vehicle has progressed through multiple design iterations, flow visualisation has been used in an attempt to identify unwanted regions of flow separation that are contributing to vehicle drag and used as a tool to guide designers in external geometric changes to minimise flow separation and hence drag. An example of these visualisations are shown in Fig. 17. The memory footprint of the simulation data (post-clustering) is approximately 4.5 GB. The clustering process takes 37 seconds.

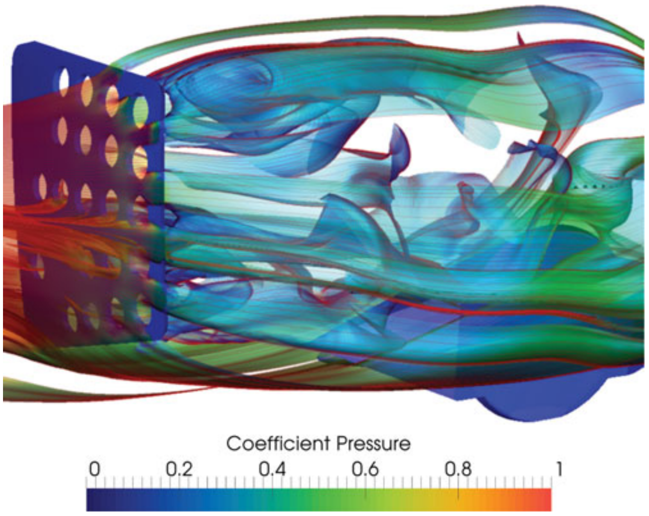

Figure 16. (Colour online) BLOODHOUND SSC CFD simulation of the the final airbrake design configuration in use, clustered using parameters $A = 0.95$, $B = 0.6$ and $k = 19$. The image is using transparency to aid the visual perception of the flow behind the airbrake. It can be seen that the flow behind this version of the airbrake is highly complex. Colour is mapped to range normalised coefficient of pressure c_p, where freestream c_p is green, high c_p is red, and low c_p is blue.

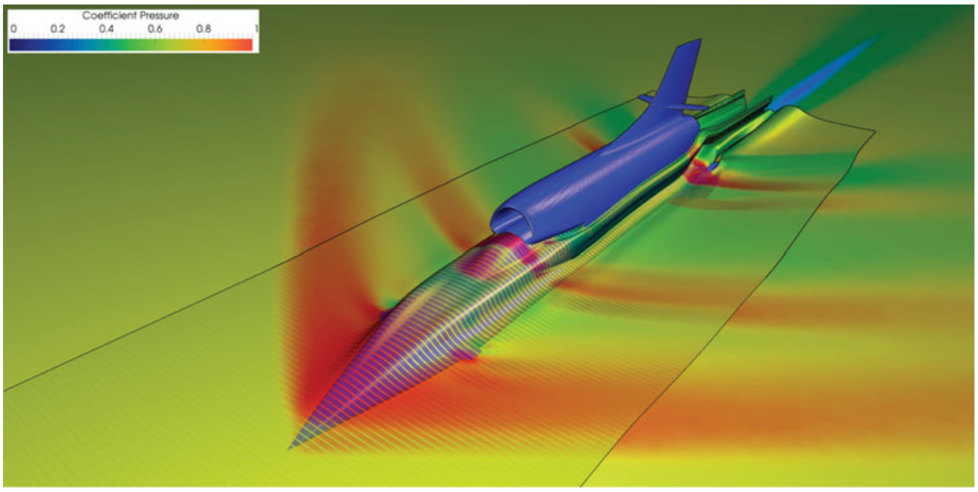

Figure 17. (Colour online) The BLOODHOUND SSC Jet and Rocket Propelled Land Vehicle. This visualisation of the CFD data at Mach 1.3 uses our algorithm to locate and seed an area of flow which curves up over the nose of the car. The seeded surface colour is mapped to range normalised coefficient of pressure c_p where red is high and blue is low relative to free stream pressure which is green. This colour mapping enables engineers to review the pressure distribution across the vehicle, assessing if the pressure distribution may cause instability during motion.

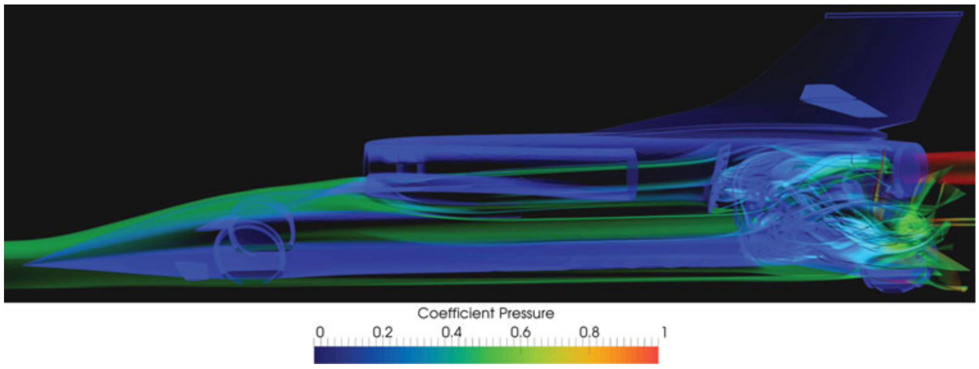

Figure 18. (Colour online) The BLOODHOUND SSC Jet and Rocket Propelled Land Vehicle. This visualisation of the CFD data at Mach 0.8 uses our algorithm to locate and seed an area of flow which curves up over the nose of the car. The seeded surface colour is mapped to the coefficient of pressure c_p where red is high and blue is low relative to free stream pressure which is green. Transparency is also used to allow the engineer to view otherwise hidden features. The stream surfaces capture the turbulent air flow in the downstream wake of the airbrakes. The surface colour mapping enables engineers to review the pressure distribution in the wake region, assessing if the pressure distribution may cause instability during motion.

5.2.1 *Full vehicle with deployed airbrakes*

During the deceleration phase, the airbrakes are deployed with the purpose of increasing drag. This additional drag increases the rate at which the vehicle decelerates. The design objective for the airbrakes is to produce large amounts of drag while maintaining steady flow. The visualisation of flow in the downstream wake of the deployed airbrakes aids the engineer in determining the quantity of turbulent fluctuations, their intensity and their distribution. These visualisation techniques support the analysis of how these fluctuating forces are transferred to the vehicle, i.e. rear suspension and body, and has helped refine and optimise the behaviour of the airbrake/vehicle interaction, minimising the impact on vehicle stability. An example of these visualisations are shown in Fig. 18.

6.0 CONCLUSIONS

This paper presents a CFD-based flow visualisation approach with improved computational performance, memory footprint, robustness of flow visualisation using stream surface placement tailored for the aerodynamic challenges encountered by the design engineers of the BLOODHOUND SSC Land Speed Record project. The performance and memory usage is improved while providing an environment and tools for the engineer to visualise undesirable flow behaviour. The approach is compared and contrasted with previous recent work in this field, providing feedback from design engineers utilising the framework who conclude that this technique is a significant improvement over recent work in this area. The choice of the initial clusters impacts the results achieved by K-means clustering after convergence to the local minimum. It is not guaranteed that the results are globally optimal. The clustering results can produce cluster centres which may reside outside the flow domain e.g. inside the object of study as a result of a cluster boundary straddling the object in Euclidean space. Our future work will examine these limitations in greater depth.

The work presented in this paper assumes either steady state solutions or uses instantaneous 'snapshots' of an unsteady flow field to create flow visualisation images. Work is ongoing to adapt the algorithm to process unsteady CFD datasets resulting in flow visualisation movies. The authors are also working on the integration of this flow visualisation approach into automated computational optimisation algorithms to provide designers with feedback even when using a 'hands-free' aerodynamic optimiser.

ACKNOWLEDGEMENTS

The authors wish to recognise the support provided by HPC Wales in the provision of computational support and to the Bloodhound Project Ltd for the support in undertaken this work. The work was also supported by the EPSRC UK Centre for Marine Energy Research (EP/M014738/1).

REFERENCES

1. BAVOLI, L. and MYERS, K. Order independent transparency with dual depth peeling, *NVIDIA Developer SDK 10*, February 2008.
2. BOCK, H.H. Clustering methods: A history of k-means algorithms, *Selected Contributions in Data Analysis and Classification*, Studies in Classification, Data Analysis, and Knowledge Organization, 2007, Springer, Berlin, Heidelberg, Germany, pp 161-172.
3. BORN, S., WIEBEL, A., FRIEDRICH, J., SCHEUERMANN, G. and BARTZ, D. Illustrative stream surfaces, *IEEE Transactions on Visualization and Computer Graphics*, 2010, **16**, (6), pp 13291338.
4. CAMARRI, S., SALVETTI, M.V., BUFFONI, M. and IOLLO, A. Simulation of the three-dimensional flow around a square cylinder between parallel walls at moderate Reynolds numbers. *Congresso di Meccanica ed Applicata*, 2005, pp 11-15.
5. CHEN, Y., COHEN, J.D. and KROLIK, J. Similarity-guided streamline placement with error evaluation, *IEEE Transactions on Visualization and Computer Graphics*, 2007, **13**, (6), pp 1448-1455.
6. EDMUNDS, M., LARAMEE, R.S., CHEN, G., MAX, N., ZHANG, E. and WARE, C. Surface-based flow visualization, *Computers & Graphics*, 2012, **36**, (8), pp 974-990.
7. EDMUNDS, M., LARAMEE, R.S., CHEN, G., ZHANG, E. and MAX, N. Advanced, automatic stream surface seeding and filtering, *Theory and Practice of Computer Graphics*, 2012, pp 53-60.
8. EDMUNDS, M., LARAMEE, R.S., MALKI, R., MASTERS, I., CROFT, T.N., CHEN, G. and ZHANG, E. Automatic stream surface seeding: A feature centered approach, *Computer Graphics Forum*, June 2012, **31**, (3.2), pp 1095-1104.
9. EDMUNDS, M., MCLOUGHLIN, T., LARAMEE, R.S., CHEN, G., ZHANG, E. and MAX, N. Automatic stream Surfaces seeding, EUROGRAPHICS 2011 Short Papers, 11-15 April 2011, Llandudno, Wales, UK, pp 53-56.
10. EITZ, M. and LIXU, G. Hierarchical spatial hashing for real-time collision detection, IEEE International Conference on Shape Modeling and Applications, 2007, Lyon, France, pp 61-70.
11. EnSight. https://www.ceisoftware.com/. Accessed: August 2014.
12. EVANS, B.J., HASSAN, O., JONES, J.W., MORGAN, K. and REMAKI, L. Computational fluid dynamics applied to the aerodynamic design of a land-based supersonic vehicle. *Numerical Methods for Partial Differential Equations*, 2011, **27**, (1), pp 141-159.
13. FANG, Y. and WANG, J. Selection of the number of clusters via the bootstrap method. *Computational Statistics and Data Analysis*, 2012, **56**, (3), pp 468-477.
14. FERREIRA, N., KLOSOWSKI, J.T., SCHEIDEGGER, C.E. and SILVA, C.T. Vector field k-means: Clustering trajectories by fitting multiple vector fields. *CoRR*, abs/1208.5801, 2012.
15. GARTH, C., KRISHNAN, H., TRICOCHE, X., TRICOCHE, T. and JOY, K.I. Generation of accurate integral surfaces in time-dependent vector fields, *IEEE Transactions on Visualization and Computer Graphics*, 2008, **14**, (6), pp 1404-1411.

16. HUMMEL, M., GARTH, C., HAMANN, B., HAGEN, H. and JOY, K.I. IRIS: Illustrative rendering for integral surfaces, *IEEE Transactions on Visualization and Computer Graphics*, 2010, **16**, (6), pp 1319-1328.
17. JOBARD, B. and LEFER, W. Creating evenly–spaced streamlines of arbitrary density, Proceedings of the Eurographics Workshop on Visualization in Scientific Computing '97, 1997, **7**, pp 45-55.
18. KOGAN, J. *Introduction to Clustering Large and High-Dimensional Data*, 2007, Cambridge University Press, Cambridge, New York, US.
19. LARAMEE, R.S., HAUSER, H., ZHAO, L. and POST, F.H. Topology-based flow visualization: The state of the art, Topology-Based Methods in Visualization Mathematics and Visualization, 2007, Springer, Berlin, Heidelberg, Germany, pp 1-19.
20. LARSSON, T., SATO, T. and ULLBRAND, B. Supercomputing in F1–unlocking the power of CFD, 2nd European Automotive CFD Conference, 2005, Frankfurt, Germany, pp 29-30.
21. LI, L. and SHEN, H.W. Image-based streamline generation and rendering, *IEEE Transactions on Visualization and Computer Graphics*, 2007, **13**, (3), pp 630-640.
22. MARCHESIN, S., CHEN, C.K., HO, C. and MA, K.L. View-dependent streamlines for 3D vector fields, *IEEE Transactions on Visualization and Computer Graphics*, 2010, **16**, (6), pp 1578-1586.
23. MATTAUSCH, O., THEUSSL, T., HAUSER, H. and GRÖLLER, E. Strategies for interactive exploration of 3D flow using evenly-spaced illuminated streamlines, Proceedings of the 19th Spring Conference on Computer Graphics, 2003, Budmerice, Slovakia, pp 213-222.
24. MENTER, F.R. Two-equation eddy-viscosity turbulence models for engineering applications, *AIAA J*, 1994, **32**, (8), pp 1598-1605.
25. MÖLLER, T. and HAINES, E. *Real-Time Rendering*, 2nd ed., 2002, A. K. Peters Limited, Natick, Massachusetts, US.
26. NOBLE, R. Bloodhound SSC. http://www.bloodhoundssc.com/. Accessed: March 2013.
27. ParaView. https://www.paraview.org/. Accessed: August 2014.
28. PEIKERT, R. and SADLO, F. Topologically relevant stream surfaces for flow visualization, Proceedings of the Spring Conference on Computer Graphics, April 2009, Budmerice, Slovakia, pp 43-50.
29. PENG, Z., GRUNDY, E., LARAMEE, R.S., CHEN, G. and CROFT, N. Mesh-driven vector field clustering and visualization: An image-based approach, *IEEE Transactions on Visualization and Computer Graphics*, 2012, **18**, (2), pp 283-298.
30. PENG, Z. and LARAMEE, R.S. Higher dimensional vector field visualization: A survey, Theory and Practice of Computer Graphics (TPCG '09), June 2009, Cardiff, Wales, UK, pp 149-163.
31. POST, F.H., VROLIJK, B., HAUSER, H., LARAMEE, R.S. and DOLEISCH, H. The state of the art in flow visualization: Feature extraction and tracking. Computer Graphics Forum, December 2003, **22**, (4), pp 775–792.
32. ROTH, M. Automatic extraction of vortex core lines and other line-type features for scientific visualization, PhD Thesis, 2000, Technische Wissenschaften ETH Zurich, Zurich, Switzerland.
33. SPALART, P.R. and ALLMARAS, S.R. A multigrid accelerated hybrid unstructured mesh method for 3d compressible turbulent flow. *AIAA Paper 92*, 439, 1992.
34. Swansea University. Swansea University College of Engineering. http://www.swansea.ac.uk/engineering/research/. Accessed: August 2013.
35. Tecplot. Tecplot 360. http://www.tecplot.com/. Accessed: March 2013.
36. TELEA, A. and van Wijk, J.J. Simplified representation of vector fields, Proceedings IEEE Visualization '99, 1999, Salt Lake City, Utah, US, pp 35-42.
37. TESCHNER, M., HEIDELBERGER, B., MÜLLER, M., POMERANETS, D. and GROSS, M. Optimized spatial hashing for collision detection of deformable objects, Proceedings of Vision, Modeling, Visualization VMV'03, 2003, Munich, Germany, pp 47-54.
38. THEISEL, H., WEINKAUF, T., HEGE, H.C. and SEIDEL, H.P. Saddle connectors–an approach to visualizing the topological skeleton of complex 3D vector fields, Proceedings IEEE Visualization '03, 2003, Seattle, Washington, US, pp 225-232.
39. TOSCHI, F. International CFD database. http://cfd.cineca.it/. Accessed: 18 March 2014.
40. VILANOVA, A., BERENSCHOT, G. and VAN PUL, C. DTI visualization with streamsurfaces and evenly-spaced volume seeding. Joint Eurographics - IEEE TCVG Symposium on Visualization, 2004, Eurographics Association, Konstanz, Germany, pp 173-182.
41. VON FUNCK, W., WEINKAUF, T., THEISEL, H. and SEIDEL, H.-P. Smoke surfaces: An interactive flow visualization technique inspired by real-world flow experiments, *IEEE Transactions on*

Visualization and Computer Graphics (Proceedings Visualization 2008), November – December 2008, **14**, (6), pp 1396-1403.
42. Weinkauf, T., Hege, H.C., Noack, B.R., Schlegel, M. and Dillmann, A. Coherent structures in a transitional flow around a backward-facing step, *Physics of Fluids*, September 2003, **15**, (9), S3.
43. Weinkauf, T. and Theisel, H. Curvature measures of 3D vector fields and their application, *J WSCG*, 2002, **10**, 2002, pp 507-514.
44. Weinkauf, T., Theisel, H., Hege, H.C. and Seidel, H.P. Boundary switch connectors for topological visualization of complex 3D vector fields, Proceedings of the Joint Eurographics – IEEE TCVG Symposium on Visualization (VisSym '04), 2004, Konstanz, Germany, pp 183-192.
45. Weiskopf, D., Schafhitzel, T. and Ertl, T. Real-time advection and volumetric illumination for the visualization of 3D unsteady flow, Data Visualization, Proceedings of the 7th Joint EUROGRAPHICS–IEEE VGTG Symposium on Visualization (EuroVis 2005), May 2005, Leads, UK, pp 13-20.
46. Wilcox, D.C. Reassessment of the scale-determining equation for advanced turbulence models, *AIAA J*, 1988, **26**, (11), pp 1299-1310.
47. Xu, R. and Wunsch, D. Survey of clustering algorithms, *IEEE Transactions on Neural Networks*, 2005, **16**, (3), pp 645-678.
48. Zöckler, M., Stalling, D. and Hege, H. Interactive visualization of 3D-vector fields using illuminated streamlines, *Proceedings IEEE Visualization '96*, October 1996, San Fransisco, California, US, pp 107-113.