

Illustrating How Hydraulic Machinery Works

submitted by

Garoe Dorta-Perez

Research Project Preparation CM50175A

Unit Leader: Prof Yong-Liang Yang

University of Bath

Department of Computer Sciences

EngD in Digital Entertainment

January 2015

COPYRIGHT

Attention is drawn to the fact that copyright of this thesis rests with its author. This copy of the thesis has been supplied on the condition that anyone who consults it is understood to recognise that its copyright rests with its author and that no quotation from the thesis and no information derived from it may be published without the prior written consent of the author.

This thesis may be made available for consultation within the University Library and may be photocopied or lent to other libraries for the purposes of consultation.

Signature of Author

Garoe Dorta-Perez

Summary

In this research proposal we present a method to semi-automatically depict *How things work* illustrations of hydraulic machinery.

Chapter 1

Introduction

How things work visualizations have been used as an efficient method to explain how a wide range of systems work. This technique usually involves displaying where each part is in relation to the system, showing how force is transmitted from one piece to the next or animating motion. In order to perform this task a range of visual transformations are used. Such as viewing the system from different angles, zoom degrees, transparency levels, as well as displaying only a subset of the parts. Generating material of this sort typically involves manual methods, usually in the form of an expert drawing each illustration by hand or composing a fixed animation using specific software.



Figure 1-1: Overview of the intended workflow

Hydraulic machinery is commonly used in our everyday lives. Such as lifting cars with jacks, rams on excavators or gerotors to control fuel intake, as shown in Figure 1-1a. Their popularity is based on their faculty to transmit a force or torque multiplication independently

of the distance between input and output. A typical hydraulic equipment has a contained liquid fluid that becomes pressurised when a force is applied to it, as shown in Figures 1-2a and 1-2b. Then that force is transmitted to the other end of the fluid. In order to grasp how the whole system works, it is essential to understand how the pressure is directed and how it interacts with other parts in the machinery. Therefore, to illustrate the general process, the spatial configuration of each part in the system must be unveiled. As well as the chain of motions that takes place within the gears and the liquid fluids.

In order to generate *How things work* visualizations, we first need a 3D model of the physical machinery we will be working with, as shown in Figure 1-1b. There are some common visualisation and illustration techniques used for hydraulic machinery, as shown in Figure 1-1c. **Motion arrows** can point out how the solid parts move and they also indicate fluid flow movement. **Frame sequences** display key frames in complex motions and they can highlight temporal interactions between the parts. **Animations** are an useful tool in highly dynamic systems, for example when an excessive number of frame sequences are needed in a particularly complicated motion.

Generating visualisations and illustrations for hydraulic machinery is challenging for designers. They must understand in detail what forces are generated as parts interact among each other and what kind of flow movements are entailed. Furthermore, when 2D illustrations are used, they main disadvantage is the impossibility to change the viewpoint to explore the object from different angle. Moreover, when animations for hydraulic visualizations are generated, they are infrequently updated. Since manual animation is a costly and time consuming task.



Figure 1-2: Sample diagrams for hydraulic mechanisms

There has been some work done on automatically generating illustrations and visualiza-

tions on mechanical assemblies. Nevertheless, it has been restricted to gear to gear interaction only. I.e. solid parts interacting with other solid parts, as gear movement can be inferred using symmetry information. Whereas work on fluid simulation and visualization has not been applied to hydraulic equipment. As simulations in this field are usually designed to, either generate complex visualizations for engineering purposes, or to produce visually plausible but not physically accurate results for animation or games.

This research proposal aims to introduce a method for generating *How things work* illustrations for hydraulic machinery. Simulations would be used to generate explanatory illustrations for hydraulic machinery. These illustrations would help users understand how this kind of equipment works.

In summary, the main contributions would be:

- An application for creating how things work illustration for hydraulic machinery 3D models.
- A method for detecting motion and interaction of fluid in the model parts.
- Algorithms to automatically generate illustrations with motion arrows and frame sequences

Chapter 2

Previous Work

2.1 Part analysis

Libraries of 3D mesh models are quite common [Tri14], [Gra14], [Aut14], in addition they contain vast quantities of models. Usually the models are also represented using points clouds, which can be obtained easily from the meshes. However, clouds and mesh models lack middle and high level information such as: symmetry, parallelism or part segmentation. Therefore we must extract this information from either representation. Since solving for general shapes is quite challenging, a number of constrained approaches have been proposed. Moreover, part analysis goes beyond part segmentation, including part semantics and part relations analysis. For an overview on segmentation techniques, we refer the readers to the following surveys [VMC97], [APP*07] and [Sha08]. Whereas, for symmetry detection we recommend [MPWC13] survey.

2.1.1 Region growing

Region growing algorithms start with a seed element for the sub-mesh and then perform a local greedy element addition to the current cluster. [MDKK06] proposed a mesh segmentation technique based on curvature estimation with sharp edge recognition. The author's method performs well on mechanical objects and it is also robust to noise. However, oversegmentation occurs in models with soft edges. Extensions to this model include Markov Random Fields [LW08],

2.1.2 Iterative clustering

Iterative clustering is based on the k-means clustering algorithm. Representatives for each cluster are chosen and then using some metric the elements are assigned to the best cluster and the representatives are recalculated. [LZHM06] presented a cluster based segmentation algorithm. The method performs better with hierarchical models composed of regions at multiple scales. Therefore, it gives the same results regardless of model scale or the coarseness of the mesh. A quadratic surface fitting algorithm was presented by [YWLY12]. Which combines several distance metrics to perform the quadratic minimization. Quadratic surfaces provide more flexibility than simpler representations, however the algorithm does not perform as well when the segments can not be represented by a quadratic surface.

2.1.3 Random walk

With the random walks approach, n faces are chosen as seeds, with n been the number of desired segments. Then for each unlabelled face, the probabilities that a random walker will reach each labelled face are calculated. Finally, each face is assigned to the highest probability label. [Gra06] proposed a first random walks implementation constrained into the imaged domain. Thereafter, [LHMR09] extended it to for 3D mesh models, the author's technique also included an automatic seeding mode where seeds quantity and placement is based on a spring-like energy function. Seeds placement can greatly affect the final segmentation. Nevertheless, the authors tackle the issue with a post processing stage where the clusters are merged if oversegmentation occurred and the edges are smoothed. Given its nature, random techniques give different results on different executions, the authors addressed the issue with their spring-like energy function, however their method is not completely invariant to seed placement.

2.1.4 Data driven

[GF09] presented an extension to the clustering technique with special emphasis on edge and face consistency, that became a new segmentation area. The algorithm is based on a graph construction which encodes triangle neighbour information. The method then performs the clustering on the graph space. The main advantage is consistent segmentation of similar models. For example, on different chair models, segments always represent legs, backs and arm-rests. However, there are several parameters that need to be input by the user. Moreover, as the method is based on consistency across a set of models, if the alignment between the different models fails, the resulting segmentation quality will decrease. Researchers have presented ex-

tensions to the original [GF09] approach, such as: setting a objective function as a Conditional Random Field model [KHS10].

2.1.5 Global energy function

[BLVD11] proposed a method based on a boundary edged function that would be learned by an AdaBoost classifier. Nevertheless, the method needs a large database of manually segmented meshes, and it also has problems generalizing the function for difficult test data. [dCdLL14] presented a novel segmentation technique based on calculating a global discrepancy function, based on the Lambert illumination model, for each triangle in the mesh. Then they directly use that value to segment the model in several parts. However, their segmentation results are still not comparable with other state of the art methods.

2.1.6 Hybrid

[WY11] approach for mesh segmentation functions with a combination of curvature estimation, Gauss mapping and B-spline surface fitting. Their methods performs relatively well, however there are six numerical parameters to adjust and the presented results show oversegmentation with certain models. [YZW14] presented a model to improve the normals calculation by adding joint information, and hence enhancing curvature detection in [WY11] method.

2.1.7 Slippage analysis

However all the previous methods cannot be directly applied to segment mechanical parts. [GG04] proposed a method for segmenting mechanical objects based on local slippage. This is quite useful as it gives for each part its degrees of freedom, e.g. sphere (3 rotations), cylinder (1 rotation and 1 translation), plane (1 rotation and 2 translations), etc. [XWY09] applied slippage analysis to perform joint-aware deformations in man-made objects. [YLT*14] improved [GG04] method making it more robust to noise and giving extra primitive information(e.g. normal of a plane, center of a sphere, etc).

2.1.8 Symmetry detection

In man-made objects, and specially with assembly models, object symmetry can be used to detect important features in a part. With symmetric parts possible axis of rotation or translation can be computed. Additionally, non symmetric ones normally imply movement constraints to the other parts.

[MGP06] presented a approximate symmetry detection method, using a voting scheme. The technique detects reflections, rotational symmetries and scaling. Mitra et al. later extended their previous work to automatic object symmetrization [MGP07]. However, parameters for symmetry detection and shape deformation still have to be manually given to the algorithm. An alternative approach to detect only rigid symmetries was proposed by [BBW*09]. Their method is based on feature lines, whose main advantage is better performance than previous methods. Nevertheless, their method is constrained to only rigid transformations. [XZJ*12] presented a partial intrinsic symmetry detection system. With a multi-scale voting scheme intrinsic symmetries are extracted on multiple scales. Anyhow, erroneous symmetry fusion between different parts may occur when the sample points are on the same scale cluster.

2.1.9 Part characterization

[GSMCO09] demonstrated that man-made objects can be deformed intuitively with a set of 1D curves extracted from the models. Variational shape approximation has also been studied for characterization purposes of engineered objects [CSAD04]. An abstraction method, to extract a characteristic curve networks was proposed by [MZL*09]. Nevertheless, Other methods has been explored as well, such as planar based ones [MSM11]. However, their approach is limited to a learning phase with a training data set manually marked. [YK12] proposed a controllable abstraction approach uses a combination of RANSAC and volume-centric primitive fitting to produce a range of simplified models.

2.2 Fluid simulation

The current paradigm in fluid simulation consist of solving the Navier-Stokes equations of fluid dynamics, shown in Equations 2.1 and 2.1.

$$\nabla \cdot \mathbf{u} = 0 \quad (2.1)$$

$$\mathbf{u}_t = -(\mathbf{u} \cdot \nabla)\mathbf{u} + \nabla \cdot (\nu \nabla \mathbf{u} - \frac{1}{\rho} \nabla p + \mathbf{f}) \quad (2.2)$$

The first equation enforces mass conservation and ensures incompressibility, while the second is derived from Newton's Second Law and encodes momentum conservation. Where \mathbf{u}_t is the time derivative vector field of the fluid velocity, p is the scalar pressure field, ρ is the density of the fluid, ν is the kinematic viscosity and \mathbf{f} represents the body force per unit mass, usually gravity.

Along the years several methods have been proposed, although all of them are based on the same Navier-Stokes equations.

- **Fourier Transform** techniques uses several superimposed sinusoidal waves to model the fluid behaviour.
- **Grid** based methods track the fluid features at fixed points in space.
- **Smooth Particle Hydrodynamics** track a large number of particle in the fluid.
- **Hybrid** algorithms are combinations of the aforementioned methods.

Regardless of the chosen method, to update the particles in the for the next frame must be computed. However, it is common for the algorithms to have a upper time step, if the update is computed after the limit there is no guaranty that the output would be reasonable.

Another important concept in fluid simulation is whether the simulation implements interplay within the fluid and rigid bodies that come in contact with it (solid-fluid coupling). And how flexible is this interaction, one way solid fluid(e.g. a rock drops on a small pool of water, so the water moves but the rock is almost not affected by the and moves it, but the fluid does not affect the rock), or fluid solid(e.g. a small buoy floating in the ocean has little effect on the surrounding water), and two way solid fluid interaction(e.g. a flexible object drops into a pool of fluid).

For more information on real time fluid simulations we refer the readers to [VLM12] and [IOS*14] surveys.

2.2.1 Fourier Transform

When simulating fluid with periodic boundary conditions, procedural simulation can be applied with a low computational cost. Usually this method is used with large masses of water (ocean simulation), as they are an approximation to periodic boundary conditions fluids. The waves can be modelled as a superimposition of sinusoidal waves, which can be efficiently decomposed using Fast Fourier Transform methods [MWM87]. Waves produced in this fashion are visually plausible but physical accuracy is not enforced inherently by the model.

Each wave has a wave number k and a wave vector \mathbf{k} . [Tes01] presented the basic technique in this area. The authors modelled the ocean surface as a summation of complex sinusoids with different wave vectors. Visually pleasing results were achieved generating random Fourier amplitudes.

The previous method was further improved by [CGG03], with the addition of solid (ships, coast line, etc). The model developed more complex wave shape and motion equations, with differentiated equations in open sea and near the shore. And by [CC06] with adaptive surface tessellation. Furthermore, the method was implemented on GPUs, and also included shallow water effects and spray dynamics. However, Fourier transform methods can still only be applied to simulate oceans or other huge masses of fluid.

2.2.2 Grid

Grid methods were among the first techniques to solve fluid simulation and so the firsts ones implement intuitive solutions. Grid based methods solve Navier-Stokes Equations 2.1 and 2.2, in a fixed position in space (i.e. grid). As fluid flows, the equations are solved in each position, giving a value for speed and pressure at each time step. This approach is advantageous as many numerical methods can be applied easily on grids and with they are more easily adaptable to GPUs implementations than other methods.

One of the firsts papers in this area introduced a Grid method [FM96] to solve Navier-Stokes equations by applying forward Euler time integration. However, their solid fluid coupling was one way only, the solids do not affect the fluid behaviour. [Sta99] extended this method in order to overcome stability issues. The author's implemented a stable solver which allowed for larger time steps, trough an operator splitting scheme and more importantly a semi-Lagrangian method to calculate the advection term.

[CMT04] proposed solid-fluid coupling algorithm for grids models using distributed La-

grange multipliers. Nevertheless, the author’s method uses a non adaptive grid and multiphase fluids and bubbles are not covered by it.

2.2.3 Smooth Particle Hydrodynamics

Smooth Particle Hydrodynamics (SPH) has been established as one of the major breakthroughs for fluid simulations in computer graphics, positioning itself as the most popular method nowadays. Compared to grid based methods that need high resolution grids to produce foam and splashes, SPH can achieve the same with less computational load. Two extra advantages are that mass conservation is automatically satisfied if the amount and mass of the particles is kept constant; and as the particles move with the fluid, the advection term does not need to be calculated explicitly.

SPH were first introduced by [Des96], where each particle encodes its position, velocity, mass and the forces that act on it.

While, [MST*04] presented a method for computing solid fluid coupling. Where boundary particles are created on the surface of the solid and then particle coupling is calculated. [AIA12] further improved the previous method with the inclusion of irregular particle distributions as well as friction and dragging. [SZMTW14] also solved stability issues in the previous SPH solid-fluid coupling techniques, using a correction scheme. Furthermore their algorithm was implemented on GPUs, thus achieving better performance than previous methods. However, the method is still not physically completely accurate and erroneous penetrations can arise in certain situations.

2.2.4 Hybrid

Hybrid methods are combinations of the aforementioned approaches. As with other hybrid techniques, the objective is to capitalize the advantages of each and discard the weaknesses.

A common hybrid approach is the particle-in-cell (PIC) method, [Har62]. It solves the advection using SPH and the incompressibility with an Euler grid. In more detail, the particles are averaged onto a grid where all the terms in the Navier-Stokes Equations 2.1 and 2.2 are calculated, except advection. Next, the particles are translated using the grid velocity. This method has been used in a variety of situations, for example [ZB05] used it for sand simulation, while [HG09] explored fire simulations. A further improvement is an implicit approach, namely the fluid implicit particle (FLIP) [JH86]. Where the particles are not updated directly

with grid data. The main strength of FLIP is that the advection term calculation is always stable. However, the particles can sometimes clamp or have voids. FLIP was extended by [RWT11] to be able to solve the equations in a coarse grid.

2.3 Flow visualization

Extensive work has been done in this area as visualizing fluid movement has a broad range of applications. However, this is a challenging task as it has to effectively display complex and copious amounts of data. Seeing that fluid simulation is generally solved by means of highly divided grids or with large number of particles, as explained in Section 2.2.

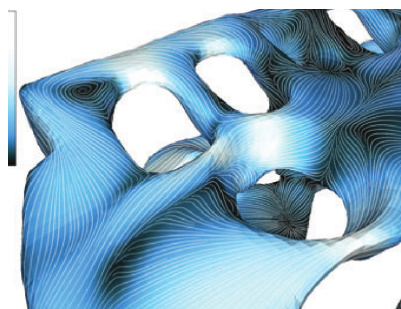


Figure 2-1: Streamlines on a 3D surface [SLCZ09].

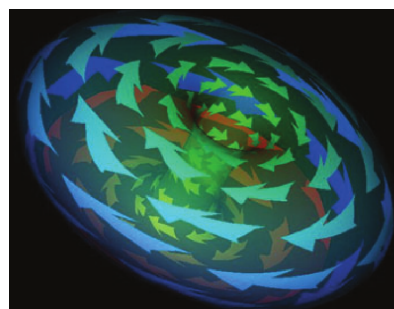


Figure 2-2: Arrows placed on streamlines paths in a 3D surface [Löf98].

Streamlines are convenient tools to describe and visualize flow, as shown in figure 2-1. A streamline is defined as a curve that is everywhere tangent to flow field, i.e. it is parallel to the local velocity vector. Therefore, they provide an intuitive mechanism to show the fluid travel direction. Furthermore, they have properties such as: streamlines will not cross each other (flow will not go across them), or a particle in the fluid starting on one streamline will not leave it. Once the streamlines has been calculated, instead of displaying them as such, they can be replaced by arrows arranged using some criteria. For instance, on the path of each streamline or after clustering some streamlines together, as shown in Figure 2-2.

In the following sections we focus only on steady flow visualization. For more information on the subject we refer the readers to [MLP*10] survey.

2.3.1 2D flow

On the 2D image domain, an image-guided algorithm for visualizing 2D flow in images was proposed by [TB96]. The approach used inefficient seeding and streamline calculations. The first problem was addressed by [JL97], with a controllable seeding algorithm using two distance parameters. While the second was solved by [MAD05], the authors used a farthest seeding point strategy with a 2D Delauney triangulation. [LHS08] presented a novel method by only generating the fewest possible number of streamlines, while also preserving the most important flow features. Exploiting local coherence, similarity between streamlines is measured and used as criteria to disregard repetitive patterns.

2.3.2 Streamlines on surfaces

Seeding techniques for curves on 3D surfaces were explored by [vW92], who developed a particle based visualization method. Particles are seeded from geometric objects and continuous points sources will create streamlines. [MHHI98] proposed a technique to generate evenly spaced streamlines, extending [TB96] work. The algorithm maps 3D surface vectors in an extended 2D image domain and maps back the streamlines to 3D. [SLCZ09] presented a method to generate streamlines only for visible parts of the surface, thus providing a significant gain in efficiency. Moreover, the technique is able to handle general surfaces, adaptive resolution and supports exploration through user interaction.

2.3.3 Streamline rendering and placement

Rendering too many streamlines can result in clutter and too few can lead to omit important characteristics. There has been plenty research into optimizing streamline production and placement. However for our hydraulic machinery the issue is less critical than in other areas.

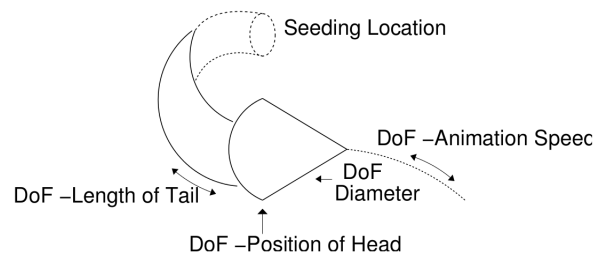


Figure 2-3: Streamcomet depiction with its degrees of freedom outlined [LH05].

[LH05] presented the streamcomet and a fast animating technique. A streamcomet, as shown in Figure 2-3, is an extension of a streamline, they move along the path of a streamline, with adjustable head position, length and translucency of tail, and variable animation speed. While the fast animating technique applies a stipple pattern to the streamline path, so the pattern is shifted at rendering time to add animation. [RPP*09] proposed a dual seeding technique for streamline placement. Dual streamlines that are orthogonal to the fluid flow are also calculated. However they are not rendered but rather used to better decide which and how the primal streamlines are going to be drawn. Considering the occlusion problems that normally arise in this area, [MCHM10] tackle the streamline rendering from a view dependant perspective. The authors precompute a random pool of streamlines. Then, occluded areas are pruned and empty areas are reseeded for the current view via occupancy buffers. What is more, a GPU implementation of their method is provided. Clustering is a common technique for reducing the amount of rendered streamlines. To address slow euclidean distance tests in clustering, [MJL*13] presented a signature based metric. The streamline attributes for the signature are curvature, torsion and tortuosity. The authors presented two variations, a fast simpler method and a hierarchical one that sacrifices speed for superior results. However, both provide a performance increase over euclidean distance calculations.

2.3.4 Surface based integral objects

Streamlines can be extended dimensionally in order to represent 3D surfaces instead of 3D lines, an example of a streamsurface is shown in Figure 2-4. Following this path leads to increased complexity in the methods, due to the new dimension needed to represent a surface. Nevertheless, surfaces can provide better visualization results. For example, the use of shading gives an improved depth insight.

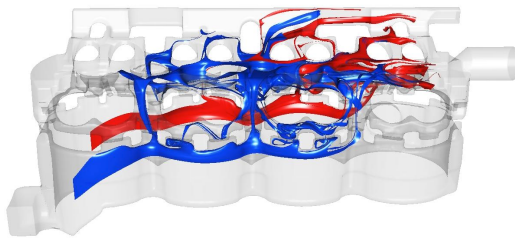


Figure 2-4: Streamsurface illustrating fuel flow in an engine [LGD*05].

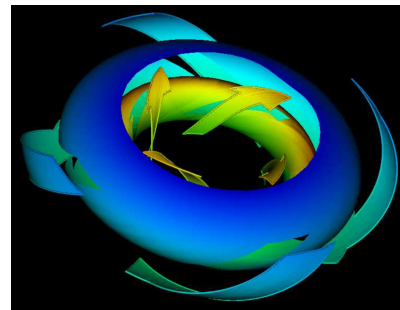


Figure 2-5: Streamarrows in an occluded flow situation [LMG97].

Streamsurfaces were introduced by [Hul92]. The author's method seeds streamlines from

a curve and those streamlines are advanced through the vector field. New streamlines are seeded or exiting ones are terminated if the points reach a predetermined maximum or minimum distance threshold. The streamlines points are used to create the streamsurface mesh. To aid in the interpretation of flow, [LMG97] extended streamsurface model with the addition of streamarrows, as shown in Figure 2-2. The author’s technique involves adding arrows to show flow direction, and removing the arrow space on the surface to help with occlusion. A hybrid method with streamlines and streamtubes was proposed by [ZDL03]. Depicting linear anisotropy regions with streamtube and planar anisotropy ones with streamsurfaces. The colors used for rendering encode additional anisotropy information. More recently, [BW10] presented extensions in streamsurface rendering to address occlusion issues, as well as a GPU implementation of their technique. Contour lines, halftoning, silhouettes, contour arrows, cuts and slabs are applied to a raw streamsurface in order to be able to depict occluded areas.

2.4 Explanatory illustration

Explanatory illustration has to adequately transmit motion on a still image, consequently transforming from the temporal space to the image domain. This is usually found in comics books and instructions sets.

2.4.1 Image and key frames output

Even though comics books are regularly considered inferior forms for presenting information, each scene contains abundant data and they can be understood by children without even reading the letters. [McC93] cited by [Cut02] reviewed visualization and abstraction techniques used in comics books. Such as, how to depict the motion of single objects and to illustrate noises and speeches bound to time. [Cut02] surveyed traditional techniques for depicting motion. Including, sequences of key frames, blur, squeeze and stretch, unbalanced postures and stroboscopic effects. The author introduced criteria to judge the effectiveness of a particular representation. For example, evocativeness, clarity of object, direction of motion and precision of motion.

[ND05] proposed a technique to depict motion in 3D animations. Scene and behaviour descriptions from specialized scene graphs were analysed in order to create the motion cues. In the author’s implementation the user has to select a depiction target and assign labels to it, using the graph information motion lines and simplified sketch images are generated. [JR05]

presented a similar approach for volumetric data. Speedlines and flow ribbons are combined to generate illustrations encoding 3D rigid movement and flow dynamics. The flow ribbons use comic inspired line abstraction technique to handle occlusion. Researchers have have look into automatic illustration generation for mechanical assemblies [MY*10]. The author’s work is based only on geometry information, extracted from 3D CAD models. Part analysis is performed using slippage analysis, see Section 2.1.7, while the motions are extracted from an interaction graph that is defined by edge contact between the components. The system is able to generate, motion arrows depicting mechanical assembly interaction, as well as generating key frames of periodic motions.



Figure 2-6: Example of arrows used to convey movement [MY*10].

2.4.2 Video output

Video based illustration is applied to enhance the motion information while maintaining the original video format. Therefore, motions are exaggerated or at the very least some hints to reinforce movement perception are added.

[CRH05] presented a method to embed cartoon-style motion cues in video. Their algorithm has two stages: a motion tracking step, where a feature manually marked will be tracked; and a computer graphics stages, where a cue will be generated and added to the video. The system can add motion lines as well as squash and stretch objects. However, the second case is restricted to videos where the background can be reconstructed from previous frames. [ZIH*11] presented a sketching system for fluids dynamics. The paper uses hybrid multi layered grid solver, discussed in Section 2.2.2, to solve 2.5D fluid flow. Flow paths and other objects can be added or changed in real time through a sketching interface. A hydraulic graph is created and is modified behind the scenes every time the visual model is edited. While the flow is shown using streamlines, discussed in Section 2.3. [Low14] showed that even though animations have become a generalized tool for visualizing dynamic systems, special care have to be taken as users can fail to extract the necessary information due to the nature of the animation.

Chapter 3

Methodology

3.1 Overview

Given a 3D CAD model of some hydraulic machinery we want to generate how things work visualizations. Namely, adding arrows depicting the fluid movement.

The problem can be subdivided into:

1. **Part analysis:** Detecting parts segments and inferring part properties.
2. **Fluid simulation:** Simulate how the fluid behaves in the previously detected parts.
3. **Flow visualization:** Display the fluid simulation data in a intuitive format.
4. **Explanatory illustration:** Transform the flow visualization data in one or several still images.

To start with, input data is needed in the form on CAD 3D models. A 3D visualization program such as SketchUp [Tri14] can be used to create, visualize and edit the models. Once we have this input, we can begin the process to generate *How things work* illustrations from them.

Part analysis involves two steps: part segmentation and part analysis. Usually 3D models consist of meshes or point clouds where there is not a clear distinction between each piece. Consequently, any additional analysis and simulation would be overly complicated without further simplifications. Therefore, a segmentation stage is needed in order to divide the model

into its constituent components. Part analysis consists of detecting the piece type, how it moves and interacts with others. So the information saved for type would be axle, gear, reservoir, fluid conduct, etc. In this classification parts are also organized considering fluid interaction criteria, e.g. if the piece interplays or not with the hydraulic fluid. With respect to types of movement, it will be direction of movement, axis of rotation, axis of translation, etc.

Once the parts have been categorized and given an input force, we will have to simulate how the force is transmitted along the different elements. In the special case where a component is a container of a fluid or is in direct contact with one, that force will have to be introduced in a fluid simulation algorithm. The output of the simulation will then carry the information along to the next piece.

In order to visualize the fluid simulation data we will need to generate a visual cue that will intuitively indicate the fluid movement. Either generating arrows indicating the overall fluid movement, with an animation or showing illustrative key frames.

Lastly, *How things work* illustrations have to be generated from the flow visualization data, that is by itself a time variant data. Therefore, techniques to embed motion in a single picture are used.

3.2 Proposed Methodology

3.2.1 Part analysis

As discussed in Sections 1 and 2, the first step is to segment the 3D CAD model mesh into useful parts for our task. In Section 2.1 we presented several options to segment meshes. However, slippage analysis [YLT*14] is the only method that can be applied directly to mechanical machinery. Moreover, this method outputs the degrees of freedom for each part. However, symmetry analysis can also be used to identify part rotation and translation axis [MGP07]. Therefore, a geometric analysis can be performed and we can work with models that do not have extra time information embedded. Which is the case for most of the models available in the model libraries in Section 2.1. However, none of the methods will automatically detect which parts will interact with the hydraulic fluid. A first approach will be to manually mark them. Nevertheless, other paths can be explored, such as heuristics that analyse the segment shape. For example, the biggest cylindrical part will probably be the reservoir for the fluid, and the parts connected to it have high probability to have the hydraulic liquid flowing through then. Other characteristics of the part have to be extracted, such as symmetry, edges, gear

radius, etc. 1D features curves are useful for characterizing man-made parts [GSMCO09].

3.2.2 Fluid simulation

Once we have segmented the model and we know how each part can move and where can the hydraulic fluid flow, we need to calculate how the liquid interacts with the solid parts. Given the characteristics of our problem, a physically accurate simulation with two way solid-fluid coupling is needed. Fourier methods are discarded because we will not be modelling vast amounts of fluid. Then, we are left with either SPH, grid or hybrid simulations. In terms of implementation complexity, grid methods are generally simpler, followed by SPH and lastly by hybrid systems. If we look at quality of the simulation, hybrid and SPH tend to be better while grid methods lag behind. However, the great disadvantage of grid simulations is the need to compute the simulation in highly divided grids in order to be able to display foams and other effects. What is more, SPH have become quite popular in the last years as a general purpose fluid simulation tool. Nevertheless, for hydraulic machinery the fluid will be constrained to a small area. Moreover, extra realistic effects, such as foams or droplets, are irrelevant for this proposal. Therefore, the previous grid disadvantage is no longer applicable, making methods such as [CMT04], the more appropriate type of simulation for our purposes.

3.2.3 Flow visualization

The next step is to generate the flow visualization from the simulation data. Following the techniques discussed in Section 2.3, we will use streamlines to visualize the flow. *How things work* illustrations do not have complex flow visualizations, as the objective is to understand roughly how the system works, not to give a perfect scientific representation of fluid dynamics. Since our CAD models are 3D, 2D methods are ruled out. There is not clear advantage in applying surface based integral objects as in hydraulic equipment there are not complex fluid flow situations. Streamcomets offer a more natural way to visualize the flow, and having a translucent body means that occlusion is less likely to happen. However, we consider streamcomets too complex for our purposes, therefore a simpler technique such as [WST09] would be used, with the optimizations presented by [MJL*13] should the generation be too slow. This flow visualization should also include an optimization step, where the arrow placement would be improved. Following the criteria in [MY*10], steps such as increasing arrow size in zones with more flow, replacing arrows in occluded areas or adding extra arrows in bigger areas, will be necessary to improve illustration quality.

3.2.4 Solid parts visualization

Solid parts visualization will be based on [MY*10] methods for mechanical assemblies. This stage involves placing arrows to convey part movement on the edge of each junction. Different arrow types will be used, such as the ones shown in Figure 2-6. Moreover, extra arrows will be added if the parts are too long and, if occlusion were to happen the arrow placement can be optimized. However, if this is not enough, the type of the arrow could be switched if it produced an increased in visibility, from cap to side or viceversa.

3.2.5 Key frames visualization

To create key frame visualizations a simple approach is to sample the animation created from the fluid simulation at key points [MY*10]. In order to select those key points there are two cases: objects with cyclical movement and parts that do not. The first ones would be spinning components, for example cylinders in engines whose movement is a periodic loop. In this case, a displacement curve can be drawn and the sample points would be the extrema and middle frames in between. In machinery that does not present cyclical movement, such as the simple cylinder shown in Figure 1-1a, it is evident that the extrema have to be included. However, we are faced with a problem of either under or oversampling in the middle points, as the minimal number of key frames to illustrate the motion are unknown.

3.3 Time line

In Figure 3-1 we present an estimated time line for the implementation of the proposal. This estimation is highly dependant on the features to be implemented and the complexity of the CAD input models. The output of one task is the input of the next, so overlapping naturally occurs when we switch to one task to the next, as we inevitably have to fix unexpected errors or introduce new features. Moreover, the changes may backtrack to tasks several steps backs as shown in the isolated bars of Part Analysis or Fluid Simulation. Furthermore, the first two tasks are given more than the rest. The first task frequently takes longer to finish, as it also includes time to set the working environment and familiarization with the tools and the area. While the second can be considered more complex than the other areas, and as such it will need extra time to reach its completion.

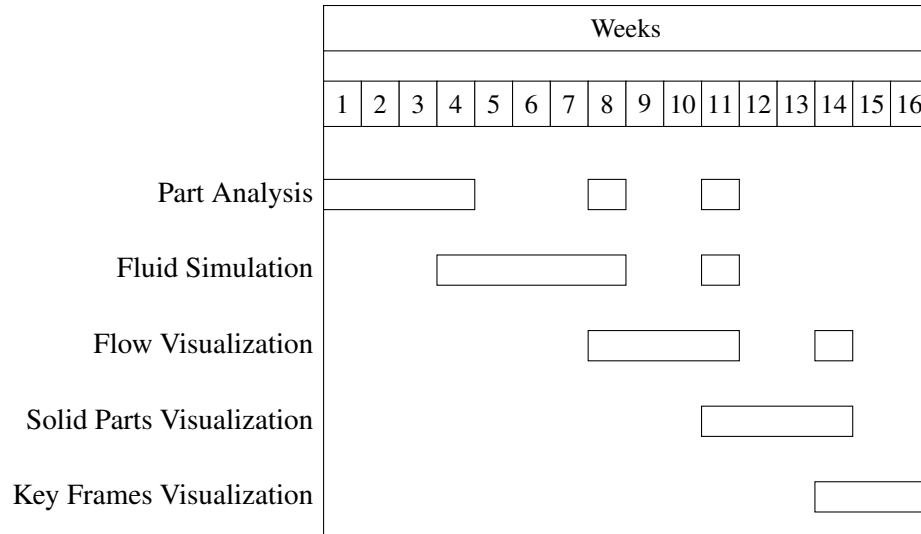


Figure 3-1: Time line for a four months completion of the proposal.

3.4 Evaluation

The system will be tested by generating illustrations for a wide variety of models. Since we do not know of any previous work which has generated *How things work* illustrations for hydraulic machinery, comparison with other methods is limited to contrasting our work against manually generated illustrations. Being able to replicate a series of chosen hydraulic illustrations could be a goal of the project. Another path to evaluate the effectiveness of the generated data is to perform user studies. One example could be an evaluation on how well the test subjects understood the machinery internal workings with a series of questions, to estimate their knowledge on the equipment before and after seeing the illustrations. Another could measure which illustration the user finds as more instructing, the automatically generated ones or manually generated, as well as asking them why they think in such a manner. Moreover, user studies can be helpful in fine tuning parameters in the algorithm, therefore improving the overall quality of the results. Comparing frame sequences, animations and single illustrations is another potential evaluation study.

Chapter 4

Applications and Conclusions

We are proposing a tool whose main aim are educational purposes. Therefore, the most obvious applications would be to generate *How things work* visualizations for museums, schools and educational centres in general. Such institutions would benefit from having a illustration generator tool. Since by its semi-automatic nature, it will address the issues with depictions created manually by experts that were discussed in Chapter 1, such as a time consuming generation stage or lack of actualization when new machinery arrives.

We have presented a proposal for a semi-automatic *How things work* illustrations generator for hydraulic machinery. The system would work with 3D CAD models of hydraulic assemblies and it would output the illustrations, either single frame or key frames. The work flow of the algorithm includes a segmentation stage, were the 3D model is divided into parts with restricted degrees of freedom, a fluid simulation stage were the hydraulic interactions are calculated, and finally a visualization stage were techniques from flow visualization and illustrating motion in single images are used to depict the overall internal workings of the hydraulic machinery.

Bibliography

- [AIA12] AKINCI N., IHMSEN M., AKINCI G.: Versatile rigid-fluid coupling for incompressible SPH. *ACM Transactions on Graphics* 31, 4 (2012), 62:1–62:8.
- [APP*07] AGATHOS A., PRATIKAKIS I., PERANTONIS S., SAPIDIS N., AZARIADIS P.: 3D Mesh Segmentation Methodologies for CAD applications. *Computer-Aided Design and Applications* 4, 6 (Jan. 2007), 827–841.
- [Aut14] AUTODESK: Autodesk 123D, 2014. URL: <http://www.123dapp.com/3d-model-library>.
- [BBW*09] BOKELOH M., BERNER A., WAND M., SEIDEL H.-P., SCHILLING A.: Symmetry Detection Using Feature Lines. *Computer Graphics Forum* 28, 2 (Apr. 2009), 697–706.
- [BLVD11] BENHABILES H., LAVOUÉ G., VANDEBORRE J.-P., DAOUDI M.: Learning Boundary Edges for 3D-Mesh Segmentation. *Computer Graphics Forum* 30, 8 (Dec. 2011), 2170–2182.
- [BW10] BORN S., WIEBEL A.: Illustrative stream surfaces. *Visualization and Computer Graphics, IEEE Transactions on* 16, 6 (2010), 1329–1338.
- [CC06] CHIU Y.-F., CHANG C.-F.: GPU-based Ocean Rendering. In *Multimedia and Expo, 2006 IEEE International Conference on* (July 2006), IEEE, pp. 2125–2128.
- [CGG03] CIEUTAT J., GONZATO J., GUITTON P.: A general ocean waves model for ship design. *Virtual Concept* 4, 7 (2003), 187–194.
- [CMT04] CARLSON M., MUCHA P. J., TURK G.: Rigid fluid: animating the interplay between rigid bodies and fluid. *ACM Transactions on Graphics (TOG)* 23, 3 (2004), 377–384.

- [CRH05] COLLOMOSSE J., ROWNTREE D., HALL P.: Rendering cartoon-style motion cues in post-production video. *Graphical Models* 67, 6 (Nov. 2005), 549–564.
- [CSAD04] COHEN-STEINER D., ALLIEZ P., DESBRUN M.: Variational shape approximation. *ACM Transactions on Graphics* 23, 3 (2004), 905–914.
- [Cut02] CUTTING J. E.: Representing motion in a static image: constraints and parallels in art, science, and popular culture. *Perception* 31, 10 (2002), 1165–1194.
- [dCdLL14] DE CASTRO P. M. M. A., DE LIMA L. A. P., LUCENA F. L. A.: Invariances of single curved manifolds applied to mesh segmentation. *Computers & Graphics* 38, 0 (Feb. 2014), 399–409.
- [Des96] DESBRUN, MATHIEU AND GASCUEL M.-P.: Smoothed particles: A new paradigm for animating highly deformable bodies. In *Computer Animation and Simulation 96*. Springer, 1996, pp. 61–76.
- [FM96] FOSTER N., METAXAS D.: Realistic Animation of Liquids. *Graphical Models and Image Processing* 58, 5 (Sept. 1996), 471–483.
- [GF09] GOLOVINSKIY A., FUNKHOUSER T.: Consistent segmentation of 3D models. *Computers & Graphics* 33, 3 (June 2009), 262–269.
- [GG04] GELFAND N., GUIBAS L. J.: Shape segmentation using local slippage analysis. In *Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing - SGP '04* (New York, NY, USA, 2004), ACM Press, pp. 214–223.
- [Gra06] GRADY L.: Random walks for image segmentation. *IEEE transactions on pattern analysis and machine intelligence* 28, 11 (Nov. 2006), 1768–1783.
- [Gra14] GRABCAD: GrabCAD, 2014. URL: <http://grabcad.com/library>.
- [GSMCO09] GAL R., SORKINE O., MITRA N. J., COHEN-OR D.: iWIRES: An Analyze-and-edit Approach to Shape Manipulation. *ACM Transactions on Graphics* 28, 3 (July 2009), 33:1–33:10.
- [Har62] HARLOW F. H.: *The particle-in-cell method for numerical solution of problems in fluid dynamics*. Tech. rep., Los Alamos Scientific Lab., N. Mex., 1962.
- [HG09] HORVATH C., GEIGER W.: Directable, high-resolution simulation of fire on the GPU. *ACM Transactions on Graphics* 28, 3 (July 2009), 41:1–41:8.

- [Hul92] HULTQUIST J. P.: Constructing stream surfaces in steady 3D vector fields. In *Proceedings of the 3rd conference on Visualization'92* (1992), IEEE Computer Society Press, pp. 171–178.
- [IOS*14] IHMSEN M., ORTHMANN J., SOLENTHALER B., KOLB A., TESCHNER M.: SPH fluids in computer graphics. In *Eurographics 2014-State of the Art Reports* (2014), no. 2, The Eurographics Association, pp. 21–42.
- [JH86] J.U. BRACKBILL, H.M. RUPPEL: FLIP: A method for adaptively zoned, particle-in-cell calculations of fluid flows in two dimensions. *Journal of Computational Physics* 65, 2 (1986), 314–343.
- [JL97] JOBARD B., LEFER W.: Creating evenly-spaced streamlines of arbitrary density. In *Visualization in Scientific Computing 97*. Springer Vienna, 1997, pp. 43–55.
- [JR05] JOSHI A., RHEINGANS P.: Illustration-inspired techniques for visualizing time-varying data. In *Visualization, 2005. VIS 05. IEEE* (2005), IEEE, pp. 679–686.
- [KHS10] KALOGERAKIS E., HERTZMANN A., SINGH K.: Learning 3D mesh segmentation and labeling. *ACM Transactions on Graphics* 29, 4 (July 2010), 102:1–102:12.
- [LGD*05] LARAMEE R. S., GARTH C., DOLEISCH H., SCHNEIDER J., HAUSER H., HAGEN H.: Visual analysis and exploration of fluid flow in a cooling jacket. In *Visualization, 2005. VIS 05. IEEE* (2005), IEEE, pp. 623–630.
- [LH05] LARAMEE R., HAUSER H.: Geometric flow visualization techniques for CFD simulation data. In *Proceedings of the 21st spring conference on Computer graphics* (2005), vol. 1, pp. 221–224.
- [LHMR09] LAI Y.-K., HU S.-M., MARTIN R. R., ROSIN P. L.: Rapid and effective segmentation of 3D models using random walks. *Computer Aided Geometric Design* 26, 6 (Aug. 2009), 665–679.
- [LHS08] LI L., HSIEH H.-H., SHEN H.-W.: Illustrative Streamline Placement and Visualization. In *Visualization Symposium, 2008. PacificVIS '08. IEEE Pacific* (Mar. 2008), IEEE, pp. 79–86.
- [LMG97] LÖFFELMANN H., MROZ L., GRÖLLER E.: Hierarchical streamarrows for the visualization of dynamical systems. In *Visualization in Scientific Computing 97*. Springer, 1997, pp. 155–163.
- [Löf98] LÖFFELMANN H.: *Visualizing Local Properties and Characteristic Structures of Dynamical Systems*. PhD thesis, Technical University of Vienna, 1998.

- [Low14] LOWE R. K.: Dynamic Visualizations: A Two-Edged Sword? In *Handbook of Human Centric Visualization*, Huang W., (Ed.). Springer New York, New York, NY, 2014, pp. 581–604.
- [LW08] LAVOUÉ G., WOLF C.: Markov Random Fields for Improving 3D Mesh Analysis and Segmentation. In *3DOR* (2008), pp. 25–32.
- [LZHM06] LAI Y.-K., ZHOU Q.-Y., HU S.-M., MARTIN R. R.: Feature sensitive mesh segmentation. In *Proceedings of the 2006 ACM symposium on Solid and physical modeling - SPM '06* (New York, NY, USA, 2006), ACM, pp. 17–25.
- [MAD05] MEBARKI A., ALLIEZ P., DEVILLERS O.: Farthest point seeding for efficient placement of streamlines. In *Visualization, 2005. VIS 05. IEEE* (2005), pp. 479–486.
- [McC93] McCloud S.: *Understanding Comics-The invisible art*. Harper Perennial, New York, NY, USA, 1993.
- [MCHM10] MARCHESIN S., CHEN C.-K., HO C., MA K.-L.: View-dependent streamlines for 3D vector fields. *IEEE transactions on visualization and computer graphics* 16, 6 (2010), 1578–86.
- [MDKK06] MIZOGUCHI T., DATE H., KANAI S., KISHINAMI T.: Segmentation of Scanned Mesh into Analytic Surfaces Based on Robust Curvature Estimation and Region Growing. In *Geometric Modeling and Processing-GMP 2006* (2006), Springer, pp. 644–654.
- [MGP06] MITRA N., GUIBAS L., PAULY M.: Partial and approximate symmetry detection for 3D geometry. *ACM Transactions on Graphics (TOG)* 25, 3 (2006), 560–568.
- [MGP07] MITRA N. J., GUIBAS L. J., PAULY M.: Symmetrization. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), Article 63.
- [MHHI98] MAO X., HATANAKA Y., HIGASHIDA H., IMAMIYA A.: Image-guided streamline placement on curvilinear grid surfaces. In *Visualization '98. Proceedings* (1998), vol. 98, IEEE, pp. 135–142.
- [MJL*13] McLOUGHLIN T., JONES M. W., LARAMEE R. S., MALKI R., MASTERS I., HANSEN C. D.: Similarity measures for enhancing interactive streamline seeding. *IEEE transactions on visualization and computer graphics* 19, 8 (Aug. 2013), 1342–1353.

- [MLP*10] McLOUGHLIN T., LARAMEE R. S., PEIKERT R., POST F. H., CHEN M.: Over Two Decades of Integration-Based, Geometric Flow Visualization. *Computer Graphics Forum* 29, 6 (Sept. 2010), 1807–1829.
- [MPWC13] MITRA N., PAULY M., WAND M., CEYLAN D.: Symmetry in 3d geometry: Extraction and applications. *Computer Graphics Forum* 32, 6 (2013), 1–23.
- [MSM11] McCRAE J., SINGH K., MITRA N.: Slices: a shape-proxy based on planar sections. *ACM Transactions on Graphics* 30, 6 (2011), 168:1–168:12.
- [MST*04] MÜLLER M., SCHIRM S., TESCHNER M., HEIDELBERGER B., GROSS M.: Interaction of fluids with deformable solids. *Computer Animation and Virtual Worlds* 15, 34 (July 2004), 159–171.
- [MWM87] MASTIN G. A., WATTERBERG P. A., MAREDA J. F.: Fourier synthesis of ocean scenes. *Computer Graphics and Applications, IEEE* 7, 3 (1987), 16–23.
- [MY*10] MITRA N. J., YANG Y.-L., YAN D.-M., LI W., AGRAWALA M.: Illustrating How Mechanical Assemblies Work. *ACM Trans. Graph.* 29, 4 (July 2010), 58:1–58:12.
- [MZL*09] MEHRA R., ZHOU Q., LONG J., SHEFFER A., GOOCH A., MITRA N. J.: Abstraction of man-made shapes. *ACM Transactions on Graphics* 28, 5 (Dec. 2009), 137:1–137:10. URL: <http://portal.acm.org/citation.cfm?doid=1618452.1618483>, doi:10.1145/1618452.1618483.
- [ND05] NIENHAUS M., DOLLNER J.: Depicting dynamics using principles of visual art and narrations. *Computer Graphics and Applications, IEEE* 25, 3 (2005), 40–51.
- [RPP*09] ROSANWO O., PETZ C., PROHASKA S., HEGE H.-C., HOTZ I.: Dual streamline seeding. In *2009 IEEE Pacific Visualization Symposium* (Apr. 2009), IEEE, pp. 9–16.
- [RWT11] RAVEENDRAN K., WOJTAN C., TURK G.: Hybrid smoothed particle hydrodynamics. In *Proceedings of the 2011 ACM SIGGRAPH/Eurographics symposium on computer animation* (2011), pp. 33–42.
- [Sha08] SHAMIR A.: A Survey on Mesh Segmentation Techniques. *Computer Graphics Forum* 27, 6 (2008), 1539–1556.
- [SLCZ09] SPENCER B., LARAMEE R. S., CHEN G., ZHANG E.: Evenly Spaced Streamlines for Surfaces: An Image-Based Approach. *Computer Graphics Forum* 28, 6 (Sept. 2009), 1618–1631.

- [Sta99] STAM J.: Stable fluids. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), pp. 121–128.
- [SZMTW14] SHAO X., ZHOU Z., MAGNENAT-THALMANN N., WU W.: Stable and Fast Fluid-Solid Coupling for Incompressible SPH. *Computer Graphics Forum* (Oct. 2014).
- [TB96] TURK G., BANKS D.: Image-guided streamline placement. In *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques* (1996), pp. 453–460.
- [Tes01] TESSENDORF J.: Simulating ocean water. *Simulating Nature: Realistic and Interactive Techniques. SIGGRAPH 1* (2001), Course Notes 47.
- [Tri14] TRIMBLE NAVIGATION LIMITED: 3D Warehouse, 2014. URL: <https://3dwarehouse.sketchup.com/>.
- [VLM12] VINES M., LEE W.-S., MAVRIPLIS C.: Computer animation challenges for computational fluid dynamics. *International Journal of Computational Fluid Dynamics* 26, 6-8 (July 2012), 407–434.
- [VMC97] VÁRADY T., MARTIN R. R., COX J.: Reverse engineering of geometric modelsan introduction. *Computer-Aided Design* 29, 4 (Apr. 1997), 255–268.
- [vW92] VAN WIJK J.: Rendering surface-particles. In *Visualization, 1992. Visualization '92, Proceedings., IEEE Conference on* (1992), pp. 54–61.
- [WST09] WICKE M., STANTON M., TREUILLE A.: Modular bases for fluid dynamics. *ACM Transactions on Graphics* 28, 3 (July 2009), 39:1–39:8.
- [WY11] WANG J., YU Z.: Surface feature based mesh segmentation. *Computers & Graphics* 35, 3 (June 2011), 661–667.
- [XWY09] XU W., WANG J., YIN K.: Joint-aware manipulation of deformable models. *ACM Trans. Graph.* 28, 3 (2009), 35:1–35:9.
- [XZJ*12] XU K., ZHANG H., JIANG W., DYER R., CHENG Z., LIU L., CHEN B.: Multi-scale partial intrinsic symmetry detection. *ACM Transactions on Graphics* 31, 6 (Nov. 2012), 181:1–181:11.
- [YK12] YUMER M. E., KARA L. B.: Co-abstraction of shape collections. *ACM Transactions on Graphics* 31, 6 (Nov. 2012), 166:1–166:11.

- [YLT*14] YI B., LIU Z., TAN J., CHENG F., DUAN G., LIU L.: Shape recognition of CAD models via iterative slippage analysis. *Computer-Aided Design* 55, 0 (Oct. 2014), 13–25.
- [YWLY12] YAN D.-M., WANG W., LIU Y., YANG Z.: Variational mesh segmentation via quadric surface fitting. *Computer-Aided Design* 44, 11 (Nov. 2012), 1072–1082.
- [YZW14] YANG X., ZHENG J., WANG D.: A computational approach to joint line detection on triangular meshes. *Engineering with Computers* 30, 4 (Dec. 2014), 583–597.
- [ZB05] ZHU Y., BRIDSON R.: Animating sand as a fluid. *ACM Trans. Graph.* 24, 3 (2005), 965–972.
- [ZDL03] ZHANG S., DEMIRALP C., LAIDLAW D. H.: Visualizing diffusion tensor MR images using streamtubes and streamsurfaces. *Visualization and Computer Graphics, IEEE Transactions on* 9, 4 (Oct. 2003), 454–462.
- [ZIH*11] ZHU B., IWATA M., HARAGUCHI R., ASHIHARA T., UMETANI N., IGARASHI T., NAKAZAWA K.: Sketch-based Dynamic Illustration of Fluid Systems. In *Proceedings of the 2011 SIGGRAPH Asia Conference* (2011), ACM, pp. 134:1–134:8.