Justin Fry CM2871

CSSE451

**Subspace Fluid Re-Simulation**

**Overview**

Subspace fluid re-simulation is a very complex and interesting topic with a number of practical applications within the graphics industry. At its most basic level, subspace fluid re-simulation consists of taking the resulting flow velocity matrix from an existing high resolution simulation, which was completed with traditional methods such as MacCormack, and produces an efficient and accurate approximation of it by using singular value decomposition for dimensionality reduction. This Approximation is then applied as the input to a cubature-based solver for Navier-stokes equations, which model fluid flow. This cubature solver allows the user to apply altered parameters and dynamics to the original simulation while maintaining the efficiency of the approximation. Examples of altered dynamics could be the buoyancy of the fluid at hand, the vorticity confinement, or the amount of laminar flow. A cubature based solver is used, as opposed to the previously used projected tensor solver, due to the fact that it allows the use of an implicit semi-lagrangian scheme for the non-linear advection function (computing a static 3rd order tensor for your advection term is possible but requires the use of finite differences and an exponential scheme, which, due to their difference from the original semi-lagrangian scheme, do not reproduce accurate re-simulations; they are however useful for efficient novel simulations). This flexibility makes it more usable for common flow modeling schemes such as the semi-lagrangian. A cubature based solver requires a cubature scheme to be precomputed. This traditionally consists of solving a non-negative least squares problem, with a matrix of cubature points chosen from each simulation snapshot multiplied by their weights shown here:



By using a probability density function to rank the likelihood of points being sampled to yield non-zero weights, and therefore non redundant data in the non-negative least squares problem, they are able to reduce fitting error and increase the efficiency of this pre-computation. Finally, by inputting the cubature scheme, desired altered dynamics and reduced velocity matrices into the cubature based subspace Navier-stokes solver, a re-simulated flow velocity matrix is generated. The result is an accurate but greatly more efficient re-simulation of flow velocities.

**Required knowledge**

The following section will briefly go over required previous knowledge that the reader may be unfamiliar with. Feel free to skip passed these if it is unnecessary.

Singular value decomposition:

The singular value decomposition(SVD) of an *m x n* matrix is its factorization into the form **M** = **UΣV**∗ where **U**  is an *m x m* unitary matrix, **Σ** is an *m x n* rectangular diagonal matrix with non-negative real numbers on the diagonal, and **V**∗,which is the transpose of **V**,is an *n x n* unitary matrix. This allows the values of M to be categorized into groups weighted by the values contained in **Σ**. To use the SVD of a matrix towards the goal of dimensional reduction, one can simply set the columns containing the smallest values in **Σ** to 0. Doing this will reduce the order of our total matrix by 1, but will have the most minimal total effect on the resultant matrix as is possible.

Semi-Lagrangian Scheme:

Throughout this summarization as well as the original paper, you will see references to schemes and how they affect the methods available for solver, most primarily, the semi-lagrangian scheme. Schemes are the characteristics that one uses to describe a fluid system, often by means of the Navier-Stokes equations. Prior to Jos Stam’s zeitgeist redefining paper introducing the semi-lagrangian scheme in the 1970s, two major schemes were in use, Eulerian schemes and Lagrangian Schemes. Eulerian schemes describe the flow velocity as v(x,t), in terms of the location and time. This can be described as standing on a river bank and looking at a spot on the river. The lagrangian description however, is like being on a boat and drifting down river and is described in terms of X(ξ,t) where ξ represents the parcel of fluid being focused on, and gives you the position of a parcel of fluid at any given time. Eulerian schemes were easy to compute but were stiff equations. To combat this problem Jos Stam introduced a semi-lagrangian scheme specific to the purpose of computer graphic simulations. It is an implicit scheme which tracks a particle’s path through a number of number of time steps, however it makes use of the eulerian grid of the eulerian scheme. This “semi-lagrangian” method produces an unconditionally stable with few downsides from the perspective of a computer graphics simulation (it does suffer from what is known as numerical dissipation, which has dampening of a flow occur to quickly, however this is negligible for graphics purposes).

**Navier-stokes**

To properly understand subspace fluid re-simulation, it is important to first understand the traditional models used to calculate the original simulation. This is usually performed by implementations of the Navier-Stokes equations which describe the motion of viscous fluid substances. These equations can be summarized as follows:



Where u is the fluid velocity, v is the viscosity constant, p is pressure, and fc is a vector of external forces. By using the semi-lagrangian scheme as created by Stam, these equations can be written as six operations. In reading order:



Where A(.) is an arbitrary advection scheme, V is a diffusion matrix, W is a velocity-to-divergence conversion matrix, X is the Poisson matrix, p is the pressure field and Y is a pressure-to-velocity conversion matrix. Prior to the developments of this cubature approach, a projected tensor method was used as a subspace solver for the Navier-Stokes equations. A divergence free basis, U, is used which eliminates the need to handle equation (2) as well as the pressure term in equation (1). The diffusion operator is fully linear so it can be directly projected onto U. The advection term, however, is non-linear, and a static 3rd order tensor with a separate discretization and exponential scheme must be created to deal with it by this method. Due to this, the subspace simulation is inconsistent with the original simulation and is therefore unfit for the purposes of re-simulation. This can be seen in the figure below with the original simulation on the left and the projected tensor simulation with the same dynamics on the right.



**On left: the original semi-lagrangian simulation. On right: the subspace projected tensor re-simulation with the same dynamics**

**Cubature approach**

In order to avoid the projected tensor method, the notion of cubature is applied to the Navier-Stokes equations. Cubature is a manner of numerically evaluating multi-dimensional algorithms, similar to Monte Carlo integration. If you have an arbitrary vector *x*, and a nonlinear function *F,* we can evaluate *F* at some point *p* by extracting rows from *x* to create *xp* and then compute *Fp* , a point-sampled version of *F* as fp=*Fp(xp)*. If done for all N points, this results in a vector f that represents x non-linearly transformed by *F*. Let us consider that our goal is to achieve a subspace evaluation of a non-linear function within the semi-lagrangian scheme to maintain consistency. We can write the subspace function as the following (U is our point sampled basis):



As stated above, cubature is a method of numerically evaluating multi-dimensional integrals, such as in (6), through summation of point sampled evaluations. We can therefore provide an accurate numerical approximation of this as a weighted sum of *F* evaluations at *P* carefully chosen samples:

****

This solves the issues encountered with the projected tensor method mention by providing a way to consistently evaluate non-linear functions after a dimensional reduction. It should be noted that while it is possible to simply project the full f vector if *F* can be written as a tensor, this method is so slow as to negate any benefits gained by the basis reduction.

**Subspace Cubature solver**

With the cubature method in place to properly evaluate non-linear functions, we can begin to describe the terms of our Navier-Stokes solver. As described in the required knowledge section, the semi-lagrangian scheme can be described by the following 6 operations:



Ignoring the non-linear terms (advection and force), the remaining stages can be easily shown performing in the subspace as follows:

****

The pressure and diffusion stages can then be combined into a single matrix-vector multiply:



The non-linear advection stage can be computed by the methods described in the previous section. By applying equation (8) to this stage we can compute the reduced quantity as the equation:



The novel force term, can be handled in two different ways depending upon the accuracy of the re-simulation desired. It can be evaluated by method of cubature, the same as the advection stage, or it can be computed in full coordinates for maximum effect.

**Multiple SVD computation**

It is noted by Kim and Delaney that if an intermediate flow value between any of the stages were to be out of the basis, it would cause error to accumulate extremely quickly. To prevent this, 4 separate U bases are constructed for the pre-advection, pre-diffusion, pre-projection, and final velocity states. These are then used to precompute the following matrices seen in the subspace Navier-Stokes stages:

****

**Dirichlet and Neumann Obstacles**

To handle obstacles that may interact with the fluid, Iterated Orthogonal Projection (IOP) is used. Recall equation (9) for the combined pressure and diffusion stages:



The IOP method imposes Dirichlet boundaries by changing the diagonal entries in I that correspond to the interior of obstacles to 0. The I with 0’s is denoted by D to obtain:



The boundaries could still be violated after the pressure solve so the process is repeated. It is noted that in the limit as iterations of the process approach infinity guarantees that the boundaries will be satisfied. However, only a few iterations are needed in practice for production purposes. To impose the Neumann boundaries, the obstacles’ velocities

are inserted into a homogeneous column appended onto D in their corresponding row. Once again this process can be repeated to insure a desired boundary condition is satisfied.

**Cubature pre-computation**

With an understanding of the subspace Navier-Stokes solver in place, we can now cover the methods used to precompute cubature points and their weights. First, a training set of T simulation “snapshots” is made available. If we have *P* promising cubature points, the weights are computed by solving a non-negative least squares problem:



The functions on the right side of the equation are computed using the full rank projected tensor method described previously, and are the projections of the example velocity fields directly post-advection. To solve this, the fit of the weights must fall within a user-defined error. If it does not, then new cubature points must be added. Kim and Delaney do this by a method they created specifically for this problem. Noting that adding many cubature candidates at a time during each pass greatly increases efficiency, they create the following algorithm:



If redundant cubature points exist, the Lawson-Hanson NNLS solver used assigns the weight to only one of them, setting the others to 0, which is why the algorithm culls points with weights equal to 0. Additionally, they use importance sampling to increase the likelihood that samples will rapidly reduce fitting error by clustering sample points in regions with large projections onto the current NNLS residual. The importance probability distribution function at each discrete fluid cell xp is defined as:



Where ap is the candidate column of A generated by xp, r is the current NNLS residual, and R is the number of points that are not yet in the cubature set. This is implemented in Algorithm 1 by replacing line 5 with Algorithm2:



**Results**

The relative efficiency of Kim and Delaney’s re-simulation method is staggering. If the external force term is computed with a full rank solve as was previously noted to be a possibility, there is an order of magnitude speedup when compared to the original simulation. If this external term computed within the subspace with the cubature method, there is a three orders of magnitude speedup when compared to the original simulation. Examples and data are as follows:

 







