Supplemental Document for Latent Space Subdivision: Stable and Controllable Time Predictions for Fluid Flow

Appendix A: Evaluation

Prediction Window Size

The prediction window w describes the count of consecutive time steps that are taken as input by the temporal prediction network. In our comparison we tested window sizes ranging from 2 over 3 up to 4 consecutive input steps. The results in terms of PSNR values are displayed in Table 5 and Table 6.

Table 5: Prediction window w comparison *Vel*

w	PSNR $\boldsymbol{u}$	PSNR ρ
4	29.67	17.04
3	29.79	16.87
2	**30.28**	**17.66**

Table 6: Prediction window w comparison *VelDen*

w	PSNR $\boldsymbol{u}$	PSNR ρ
4	**29.98**	**18.24**
3	29.52	17.84
2	29.11	17.12

It becomes apparent that the prediction-only approach (*VelDen*) benefits from a larger input window, whereas the *Vel* approach with reinjected external information performs best with a smaller input window.

Latent Space Split Percentage

We evaluated the impact of the latent space split percentage on three of our datasets. Therefore, we trained multiple models with different split percentages on the individual datasets. The comparison for our moving smoke scene is shown in Table 7 and Table 8. The latter are the results of the prediction-only evaluation (denoted *VelDen*), whereas the first table presents the results of our reinjected density approach (denoted *Vel*). In this experiment all split versions are outperformed by the No-Split version in the prediction-only setup with PSNR values of 29.71 and 18.03 for velocity and density, respectively.

Table 7: LS split comparison *Vel*; moving smoke

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
0.33	28.07	15.84
0.5	29.28	16.49
0.66	**30.28**	**17.66**

Table 8: LS split comparison *VelDen*; moving smoke

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
No-Split	**29.71**	**18.03**
0.33	28.49	16.63
0.5	29.06	17.38
0.66	29.11	17.12

Table 9: LS split comparison *Vel*; rotating cup

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
0.33	36.67	28.46
0.5	36.66	29.22
0.66	**38.52**	**29.73**

Table 10: LS split comparison *VelDen*; rotating cup

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
No-Split	**37.90**	22.68
0.33	37.52	25.01
0.5	36.77	25.13
0.66	37.57	**25.32**

Table 11: LS split comparison *Vel*; rotating and moving cup

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
0.33	35.67	25.10
0.5	**36.94**	**26.59**
0.66	36.50	26.25

Table 12: LS split comparison *VelDen*; rotating and moving cup

LS Split	PSNR $\boldsymbol{u}$	PSNR ρ
0.33	**37.89**	**26.45**
0.5	37.30	26.16
0.66	37.56	26.14

Table 13: No-Split and LSS comparison; rotating cup; 100 time steps

LS Split	Type	PSNR $\boldsymbol{u}$	PSNR ρ
No-Split	*VelDen*	37.90	22.68
0.66 (ours)	*VelDen*	37.57	25.32
0.66 (ours)	*Vel*	**38.52**	**29.73**

In contrast, the networks trained on the rotating cup dataset behave different as shown in Table 9 and Table 10. The classic No-Split version is outperformed by all other split versions in terms of density PSNR values in the prediction-only (*VelDen*) setup. In the reinjected density evaluation (*Vel*), the benefit of latent space splitting becomes even more apparent when comparing the PSNR values of velocity, 38.52 and density, 29.73 of the 0.66 network with the velocity PSNR of 37.90 and density PSNR 22.68 of the No-Split version.

No-Split vs. Latent Space Subdivision

In this section we present additional results for our rotating cup dataset. See the main document for a long-term comparison of No-Split vs. LSS for our more complicated moving and rotating cup dataset. In Table 13 we compare the temporal prediction performance of a No-Split version and our 0.66 LSS version over a time horizon of 100 simulation steps. Our LSS 0.66 version with a density PSNR value of 29.73 clearly outperforms the No-Split version with a density PSNR value of 22.68.

Generalization

Additionally, we show in Figure 15 that our method recovers from the removal of smoke in a certain sink-region and is capable of predicting the fluid motion.

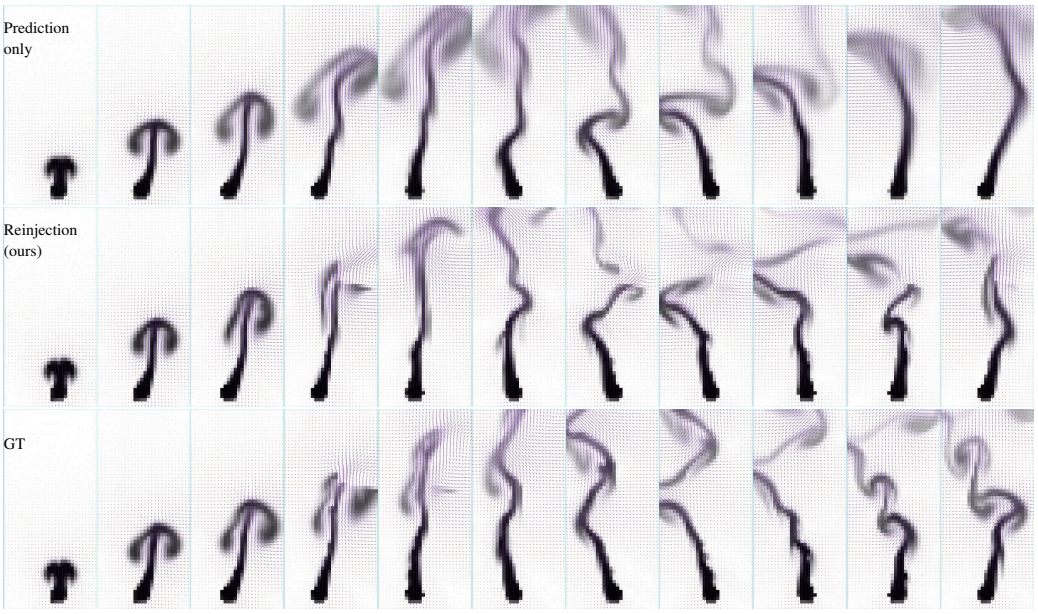

Figure 15: An sink is placed in the upper right of our moving smoke scene. This was unseen during training. The prediction by our proposed method remains stable and realistic. In the second row density reinjection was applied. In the top row no external information was injected. Thus, the sink can't be processed by the network.

Appendix B: Fluid Flow Data

Our work concentrates on single-phase flows, modelled by a pressure-velocity formulation of the incompressible Navier-Stokes equations as highlighted in Section 2. Thereby, we apply a classic NS solver to simulate our smoke flows based on R. Bridson [Bri15]. In addition to Section 4, more information about the simulation procedure is provided in the following.

Simulation Setup

The linear system for pressure projection is solved with a conjugate gradient method. The conjugate gradient (CG) solver accuracy is set to $1 \cdot 10^{-4}$ for our moving smoke dataset, whereas an accuracy of $1 \cdot 10^{-3}$ is utilized for the moving cup datasets. We generated all our datasets with a time step of 0.5. Depending on the behavioral requirements of our different experiments with rising, hot and sinking, cold smoke we use the Boussinesq model with the smoke density in combination with a gravity constant of $(0.0, -4 \cdot 10^{-3}, 0.0)$ for the moving and rising smoke and $(0.0, 1 \cdot 10^{-3}, 0.0)$ for the rotating cup dataset. To arrive at a more turbulent flow behavior, the gravity constant was set to $(0.0, 1 \cdot 10^{-2}, 0.0)$ for our moving and rotating cup dataset. We do not apply other forces or additional viscosity. We purely rely on numerical diffusion to introduce viscosity effects.

In combination with the quantities required by our classic NS setup, namely flow velocity $\boldsymbol{u}$, pressure p and density ρ, we also need a flag grid f, an obstacle velocity field $\boldsymbol{u}_{obs}$ and the corresponding obstacle levelset for our obstacle supporting scenes. Thereby our density ρ is passively advected within the flow velocity $\boldsymbol{u}$.

To handle the obstacle movement accordingly, we calculate the obstacle velocity field by evaluating the displacement per mesh vertex of the previous to the current time step and applying the interpolated velocities to the according grid cells of the obstacle velocity field. Afterwards, the obstacle velocity field values are averaged to represent a correct discretization.

In Algorithm 1 the simulation procedure of the moving smoke dataset is shown. For our obstacle datasets the procedure in Algorithm 2 is used, with the prediction algorithm given in Algorithm 3. Boundary conditions are abbreviated with BC in these algorithm.

Algorithm 1 Moving smoke simulation

while $t \rightarrow t+1$ **do**
  $\rho \leftarrow$ applyInflowSource(ρ, s)
  $\rho \leftarrow$ advect(ρ, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ advect($\boldsymbol{u}$, $\boldsymbol{u}$)
  $f \leftarrow$ setWallBCs(f, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ addBuoyancy(ρ, $\boldsymbol{u}$, f, $\boldsymbol{g}$)
  $p \leftarrow$ solvePressure(f, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ correctVelocity($\boldsymbol{u}$, p)
end while

Training Datasets

In Figure 16, Figure 17 and Figure 18 multiple simulations contained in our training data set are displayed.

Algorithm 2 Rotating and moving cup

while $t \rightarrow t+1$ **do**
  $\rho \leftarrow$ applyInflowSource(ρ, s)
  $\rho \leftarrow$ advect(ρ, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ advect($\boldsymbol{u}$, $\boldsymbol{u}$)
  $\boldsymbol{u}_{obs} \leftarrow$ computeObstacleVelocity($obstacle^t$, $obstacle^{t+1}$)
  $f \leftarrow$ setObstacleFlags($obstacle^t$)
  $f \leftarrow$ setWallBCs(f, $\boldsymbol{u}$, $obstacle^t$, $\boldsymbol{u}_{obs}$)
  $\boldsymbol{u} \leftarrow$ addBuoyancy(ρ, $\boldsymbol{u}$, f, $\boldsymbol{g}$)
  $p \leftarrow$ solvePressure(f, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ correctVelocity($\boldsymbol{u}$, p)
end while

Algorithm 3 Rotating and moving cup network prediction *Vel*

while $t \rightarrow t+1$ **do**
  $\rho \leftarrow$ applyInflowSource(ρ, s)
  $\rho \leftarrow$ advect(ρ, $\boldsymbol{u}$)
  $\boldsymbol{u} \leftarrow$ advect($\boldsymbol{u}$, $\boldsymbol{u}$)
  $\boldsymbol{u}_{obs} \leftarrow$ computeObstacleVelocity($obstacle^t$, $obstacle^{t+1}$)
  $f \leftarrow$ setObstacleFlags($obstacle^t$)
  $f \leftarrow$ setWallBCs(f, $\boldsymbol{u}$, $obstacle^t$, $\boldsymbol{u}_{obs}$)
  $\dot{\boldsymbol{c}}^t \leftarrow$ encode($\tilde{\boldsymbol{u}}^t$, ρ^t)
  $\hat{\boldsymbol{c}}^t \leftarrow [\tilde{\boldsymbol{c}}^t_{vel}, \dot{\boldsymbol{c}}^t_{den}]$
  $\tilde{\boldsymbol{c}}^{t+1} \leftarrow$ predict($\hat{\boldsymbol{c}}^{t-1}$, $\hat{\boldsymbol{c}}^t$)
  $\tilde{\boldsymbol{u}}^{t+1}, \tilde{\rho}^{t+1} \leftarrow$ decode($\tilde{\boldsymbol{c}}^{t+1}$) ▷ $\tilde{\rho}^{t+1}$ is not used
  $\boldsymbol{u}^{t+1} \leftarrow \tilde{\boldsymbol{u}}^{t+1}$ ▷ overwrite the velocity with the prediction
end while

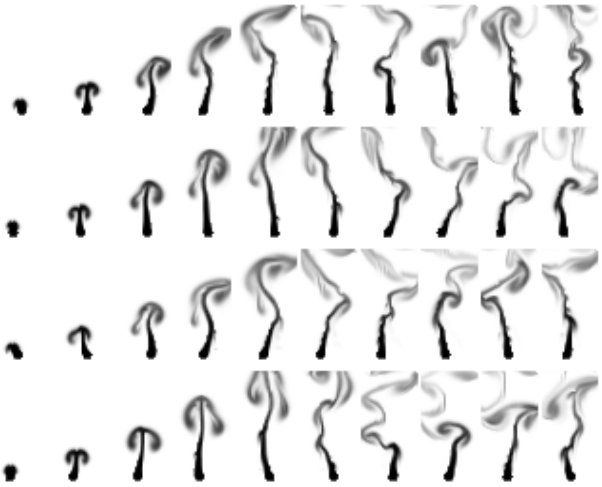

Figure 16: Four example sequences of our moving smoke dataset. For visualization purposes we display frames 20 to 200 with a step size of 20 for the respective scenes. The smoke density is shown as black.

Figure 17: Three example sequences of our rotating cup dataset. For visualization purposes we display frames 40 to 180 with a step size of 20 for the respective scenes. The cup-shaped obstacle is highlighted in blue, whereas the smoke density is shown as black.

Figure 18: Three example sequences of our rotating and moving cup dataset. For visualization purposes we display frames 40 to 180 with a step size of 20 for the respective scenes. The cup-shaped obstacle is highlighted in blue, whereas the smoke density is shown as black.