CSE 291 - Differentiable Programming - Project Report - Implement paper "Fluid Control Using the Adjoint Method implementation"

Name: Long-Giang Vu
PID: A69031012

Contents

1 Implementation Task

1.1 Tentative Tasks

- Implement the adjoint method in Loma.
- Visualize simulation results.
- Implement checkpointing to reduce memory usage.

1.2 Current Progress

- Completed the implementation of fluid operations and successfully differentiated them using Loma's reverse-mode automatic differentiation. The operations include: `APPLYCONTROL` (for applying control parameters via a linear map), `ADVECT`, `DIFFUSE`, `HEAT`, `PROJECT`, `REDISTANCE`, and `MATCHKEYFRAME`.
- Used the computed gradients to update control parameters.
- Visualized a 2D smoke simulation with a resolution of 40×40. The sequence of operations in a single timestep is: `APPLYCONTROL` $\rightarrow$ `ADVECT` $\rightarrow$ `DIFFUSE` $\rightarrow$ `HEAT` $\rightarrow$ `PROJECT` $\rightarrow$ `MATCHKEYFRAME`.

1.3 Incomplete Tasks

- Checkpointing has not yet been implemented.
- Currently, only one simulation result with smoke material has been produced.

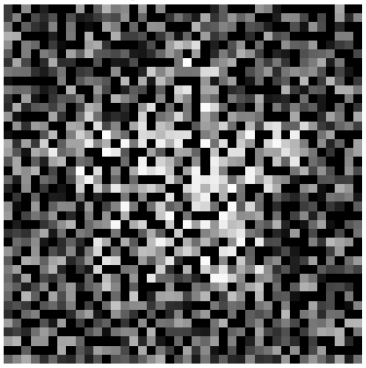
2 Challenges

While working on this project, I encountered several challenges:

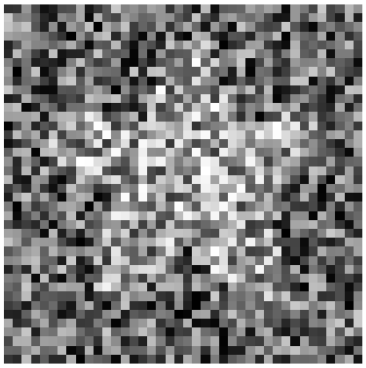
- Loma lacks native support for matrix operations such as matrix multiplication and dot products. As a result, I had to manually implement matrix multiplication using `while` loops.
- Loma also does not support common built-in functions like sorting, and it requires static memory allocation. This constraint made it difficult to implement algorithms that rely on dynamic structures, such as search.
- Reverse-mode differentiation in Loma is sensitive to function complexity. For example, the `REDISTANCE` function calls `fast_matching` (or `solve_eikonal`), which becomes too complex to debug in its differentiated form. To resolve this, I manually unrolled the logic of `fast_matching` into the main function body. This introduced redundancy but allowed the reverse-mode differentiation to function correctly.

Adjoint State 0 vs Immediate State 0

Adjoint/Solved 0

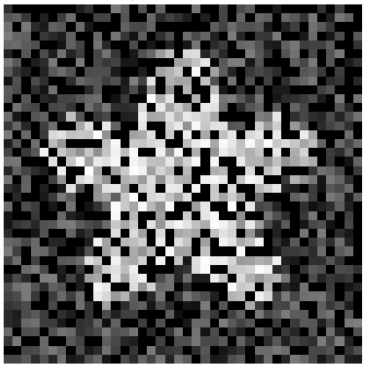


Immediate 0

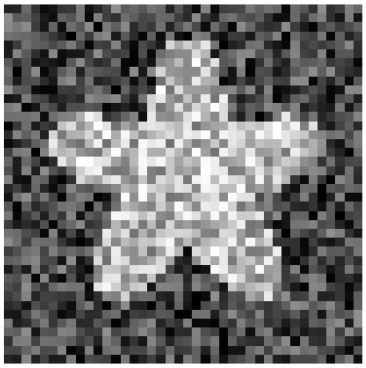


Adjoint State 1 vs Immediate State 1

Adjoint/Solved 1

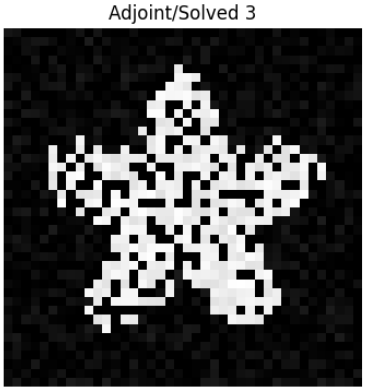


Immediate 1

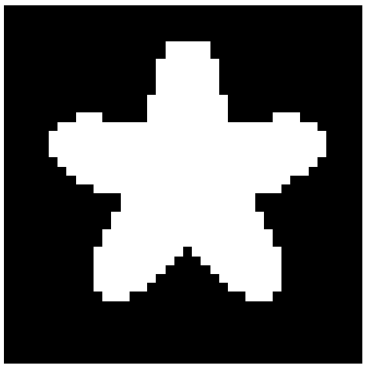


Adjoint State 3 vs Desired State

Adjoint/Solved 3



Desired



3 Result

The data of these states are stored in data folder in my code folder submission.