This document provides step-by-step instructions to build and run the Navier-Stokes solver for surface meshes with boundaries.

## 1 System Requirements

Ensure that your system meets the following requirements:

- **Operating System:** Linux, macOS
- **Build System:** CMake .
- **Graphics Libraries:**
  - OpenGL
  - GLFW

## 2 Building the Project

To compile the Navier-Stokes solver, follow these steps:

1. **Navigate to the project directory.** Open a terminal and move to the root folder of the repository:

   ```
   cd path/to/NavierStokesSolver
   ```

2. **Run the configuration script.** Execute the configuration script to set up the build environment and check for dependencies:

   ```
   ./configure.sh
   ```

   This step ensures that all necessary paths, compiler options, and dependencies (such as OpenGL, Eigen, and TinyBLAS) are correctly set up.

3. **Compile the project.** Use the provided script to compile the source code:

   ```
   ./make.sh
   ```

   This step invokes the compiler and links all necessary libraries, generating the final executable binary.

## 3 Running the Program

After successfully compiling the project, the Navier-Stokes solver can be executed with different domain configurations using the following general command format:

```
./build/release/test_NS <domain> <resolution>        (1)
```

where:

- `<domain>` specifies the computational domain for the simulation.
- `<resolution>` is a non-negative integer that determines the resolution of the mesh.

### 3.1 Available Domains

The following table lists the predefined domains that can be used:

| **Domain** | **Description** |
|---|---|
| `cube` | A cubic structured mesh |
| `sphere` | A spherical mesh derived from a cube |
| `hemisphere` | The upper hemisphere obtained by truncating a sphere |
| `2Dsquare` | A structured 2D square grid mesh |

Table 1: List of available domains for simulation.

### 3.2 Example Commands

To run the solver with different domains, use the following examples:

```
# Run on a Cube Mesh
./build/release/test_NS cube 20
```

Here, the number `20` represents the resolution parameter and can be replaced with any non-negative integer depending on the desired mesh refinement.

## 4 Code Structure

- `src/mesh/` → Mesh generation & handling.
- `src/navier_stokes/` → Navier-Stokes solver implementation.
- `src/utils/` → Utility functions.
- `shaders/` → OpenGL shaders for visualization.
- `build/` → Compiled binaries and CMake cache.

## 5 Customizing the Solver

### 5.1 Modifying the Initial Conditions

Edit the equation for the initial vorticity in `test_navier_stokes.cpp`:

```
char rhs_expression[128] =
    "100 * z * exp(-50*z^2) * (1 + 0.5 * cos(20 * theta))";
```

Change this expression to define different initial vorticity distributions.