Gas dispersion within an urban street canyon – a CFD code comparison

Magnus Aarskaug Rud and Carl Erik Wasberg

Norwegian Defence Research Establishment (FFI)

1st February 2016

FFI-rapport åårr/yyyyy

xxx

P: ISBN
E: ISBN

Keywords

Neutral gas

Large eddy simulation

Urban street cayon

Fluent performance test

Fluent vs. CDP

Fluent solver settings

Approved by

(prosjektleder)	Project manager
(avdelingssjef)	Director
John Mikal Størdal	Director General

English summary

This report investigates how well the commercial CFD software Fluent can estimate dispersion of a neutrally buoyant gas. The results obtained are compared with data from a wind-tunnel experiment and similar simulations performed in CDP. All simulations are performed using Large Eddy Simulation (LES). It is also experimented with different discretization schemes available in Fluent, the effect of a subgrid-scale model as well as confirming that the accuracy increases with the degrees of freedom. The reference case used to compare the data is gas dispersion of a neutral gas in a wind-tunnel with four cubic obstacles representing an urban street canyon.

In city planning and large projects within infrastructure it is important to have knowledge of how hazardous gases and liquids should be handled. Being able to simulate these scenarios enable us to be better prepared for unwanted events such as a factory gas leak. The choice of software and mathematical formulation is important in order to provide a correct solution within a reasonable amount of time.

The results obtained using Fluent agree well with the experimental data and the CDP simulations, The width and height of the plume generated by the gas release is consistent with the previous results. There are some disagreements in the data measured close to the wall. The choice of discretization scheme has a significant impact on the computational time and the results vary to some degree. Allthough the time can be saved by applying a non-default scheme the results indicate that the default setting in Fluent provides the most accurate solution. The simulations have been performed on up to 120 cores and the program scales very well as the amount of cores is increased.

Sammendrag

Denne rapporten undersøker hvor godt den kommersielle CFD-programvaren Fluent kan estimere dispersjon av en nøytral gass. Resultatene er sammenlignet med data fra et vindtunnel-eksperiment og tilsvarende simuleringer i CDP. Alle simuleringer er utført ved hjelp av Large Eddy Simulations (LES). Det er også eksperimentert med ulike diskretiseringsskjemaer som ligger tilgjengelig i Fluent, effekten av en subgrid-scale (SGS) modell og bekreftelse av at nøyaktigheten øker med antall frihetsgrader. Testproblemet som er brukt for sammenligning av data er dispersjon av en nøytral gass i en vindtunnel med fire kubiske klosser som representerer et urbant bygningsmiljø.

I byplanlegging og større prosjekter innenfor infrastruktur er det viktig å ha kunnskap til hvordan farlige gasser og væsker bør behandles. Å kunne simulere disse scenarioene tillater oss å være bedre forberedt på uønskede hendelser som gasslekkasje fra en fabrikk. Valg av programvare og matematisk formulering er viktig for å kunne tilby en korrekt løsning innenfor en akseptabel tidsramme.

Resultatene fra Fluent stemmer godt overens med de eksperimentelle dataene og simuleringene gjort i CDP. Bredden og høyden på gass-skyen som oppstår som en følge av gassutslippet er konsistent med tidligere data. Ulike diskretiseringsskjemaer har et signifikant utslag på regnetiden og resultatene varierer også i noen grad. Selv om tid kan bli spart ved å anvende andre innstillinger enn de som er anbefalt av Fluent, tilsier resultatene at standardinstillingene i Fluent gir de mest nøyaktige resultatene. Simuleringene har vært gjennomført på opptil 120 kjerner og koden skalerer godt etterhvert som antall kjerner øker.

Contents

1 Introduction

The scenario investigated in this work is dispersion of a neutral gas in a rectangular tunnel with four cubic blocks placed as obstacles. The blocks have sides $h = 0.109$ m and represent a set of buildings forming a street canyon. The gas is released from a circular source on ground level and is translated by the wind field through the canyon, see figure 1.1. In this figure h is used as the length scale. The dotted lines indicate the positions where data is collected.

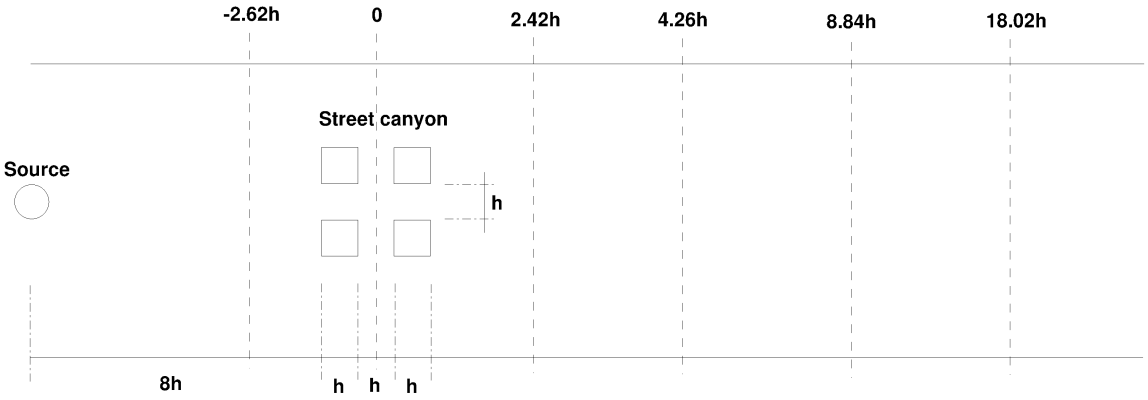

Figure 1.1 Schematic overview of the domain from above. The data is collected along the dotted lines.

Scaling the domain with the size of the boundary layer $H = 1$ m restricts it to the box $1.04 \leq x/H \leq 6, -1.75 \leq y/H \leq 1.75, 0 \leq z/H \leq 1.5$. The four cubic boxes are centered around $(2.4715, 0)$ with a distance h between each box. The source is placed with its center in $(1.436, 0)$ and radius $r = 0.0515$. The grid used for the simulation consists of 6.2 million nodes and is described in detail in [**?**]. It is an unstructured hex-mesh that can be divided into three regions (R1,R2,R3) with different spatial discretization. The finest region is concentrated around the area where the plume is assumed to be located. In addition the grid size defining the source is even finer than in the rest of R1. The grid sizes in wall units, based on the length scale $l^+ = lU^*/\nu$ is specified in table 1.1. The variables ν, U^* are defined in table 1.2. An illustration of the three regions is found in figure 1.2 The simulations are performed using Large Eddy Simulation (LES) with the dynamic Smagorinsky-Lilly subgrid-scale model [**?**]. The release of gas will result in a plume that is advected with the wind field. The size and shape of the plume at the indicated positions in figure 1.1 are compared with experimental data and simulations performed in CDP. The wind-field in the tunnel is created by an inflow condition that is defined from previous simulations in CDP [**?**]. For clarification, some of the variables used a lot throughout this work will be

Region	Scaled grid size			Grid size in mm		
	Δx^+	Δy^+	Δz^+	Δx	Δy	Δz
Source(R1)	6.3	6.3	4.0 - 13	1.5	1.5	1 - 4
R1	13	13	4.0 - 13	4	4	1 - 4
R2	61	61	20 - 61	13	13	3 - 17
R3	185	185	60 - 185	45	45	10 - 50

Table 1.1 Computational grid size used for the street canyon simulation. Δx^+ and Δy^+ are constant in each region.

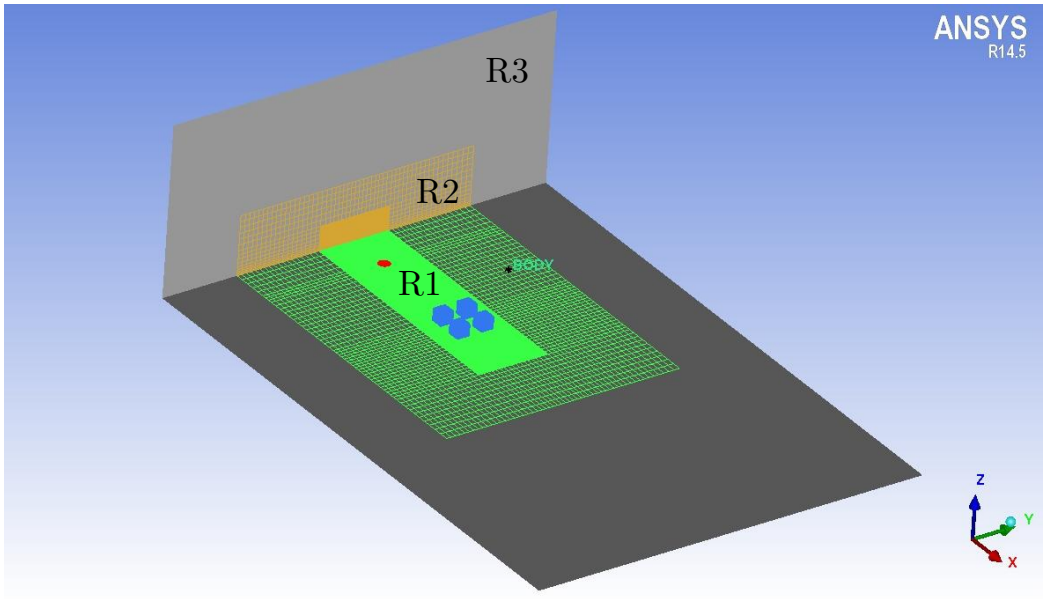

Figure 1.2 Overview of the three refinement regions

stated explicitly in table 1.2.

1.1 CDP vs. Fluent

Fluent is a widely used piece of commercial CFD software with many different solvers and discretization schemes available. CDP is a software surging from Stanford University's Center for Integrated Turbulence Simulations (CITS), Stanford, California. It is a Finite Volume (FV) based solver specially designed for LES. Fluent is also a FV based solver but there are some conceptual differences between the two codes. This section contains a brief introduction into how the numerical solvers are implemented in the two pieces of softwares.

For the main results in this work, Fluent is used with the default settings for Large Eddy Simulations of viscous incompressible flow with species transport. The algorithm used in Fluent is a pressure-velocity coupling scheme called SIMPLE. It is a predictor-corrector method for solving the equations of continuity Eq. (2.1) and momentum Eq. (2.2) and is documented in the Fluent Theory Guide [**?**]. This scheme provides estimations for the

Variable	Value	Unit	Description
H	1	m	Length scale of the domain
h	0.109	m	The sides of the cubic boxes
Q	50	dm^3/min	Gas release from source
U_{ref}	≈ 1.08	m/s	Reference value of U
U^*	≈ 0.06	m/s	Friction velocity
ν	$1.7895 \cdot 10^{-5}$	m^2/s	Kinematic viscosity

Table 1.2 Essential variables, U_{ref} is calculated as a time average of the velocity in x-direction at a point far away from the floor and walls and will therefore vary by a small amount from case to case.

velocity $\mathbf{u}$ and the pressure p for each timestep. When these estimations are obtained the transport equation for the species transport is solved. This procedure is continued until convergence is reached within each timestep.

The numerical method used in CDP VIDA is a predictor-corrector method developed by Ham (2007) [**?**] and has a slightly different approach. Instead of first solving for pressure and velocity VIDA makes an initial prediction of the flux and the mass fraction of the species. This allows it to calculate the density. The prediction for flux is then corrected by conserving mass, and the mass fraction is corrected by the transport equation. After doing these calculations the velocity and pressure equations are solved in a coupled manner similar to the way Fluent does it. The method is described in *Cascade's User's & Developer's manual 2014, chap. 9.2* [**?**].

The most important difference is that CDP VIDA solves an extra Poisson equation for each iteration and is therefore more stable and achieves faster convergence, whereas the algorithm applied in Fluent requires more iterations per timestep and is more dependent on its stability coefficients.

2 Mathematical formulation

Assuming incompressible flow, the governing equations are the continuity equation,

$$\nabla \cdot \mathbf{u} = 0, \tag{2.1}$$

and the Navier-Stokes (N-S) equation obtained by conserving momentum,

$$\rho \frac{\partial \mathbf{u}}{\partial t} + \rho \mathbf{u} \cdot \nabla \mathbf{u} = \rho \mathbf{g} - \nabla p + \frac{\partial}{\partial x_j} \left[2\mu S_{ij} - \frac{2}{3} \delta_{ij} \frac{\partial u_j}{\partial x_i} \right]. \tag{2.2}$$

In these equations $\mathbf{u} = [u_1, u_2, u_3]$ is the velocity of the flow, p is the pressure, ρ is the mass density of air, $S_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$ is the strain rate tensor, $\mathbf{g} = [0, 0, -g]$ is the gravitational acceleration, μ is the dynamic viscosity of air and δ_{ij} is the Kronecker delta function. Since

the gas released has the same density as air, the gravitational term is not taken into account in these calculations. In order to compute the transport of the gas an additional equation has to be solved, namely the transport diffusion equation for the mass fraction Z of the neutral gas,

$$\frac{\partial}{\partial t}(\rho Z) + \nabla \cdot (\rho \mathbf{u} Z) = -\nabla \cdot \mathbf{J} + S. \tag{2.3}$$

The flux of the species is denoted $\mathbf{J}$ and the source is denoted S. Note that this formulation is valid for heavy gas as well, the big difference being that the mass density ρ will be constant for the neutral gas dispersion, while for heavy gas it will be a function of Z. For the simulations done with heavy gas the volume-weighted mixing law is applied,

$$\rho(Z) = \frac{1}{\frac{Z}{\rho_{co_2}} + \frac{1-Z}{\rho_{air}}}. \tag{2.4}$$

2.1 Turbulence modelling

The governing equations are solved using LES, which consists of simulating the large scale structures and modelling the smaller ones. The filter width determines the size of the structures to be resolved and of those to be modelled. For this work the grid size is used as the filter width. The grid size is the lowest possible bound and a common choice for the filter width. The filtered N-S equation is then given as

$$\rho \frac{\partial \tilde{\mathbf{u}}}{\partial t} + \rho \tilde{\mathbf{u}} \cdot \nabla \tilde{\mathbf{u}} = \rho \mathbf{g} - \nabla \tilde{p} + \nabla \cdot \left[2\mu \mathbf{S} - \frac{2}{3} \mathbf{I} \nabla \tilde{\mathbf{u}} \right] + \nabla \cdot \boldsymbol{\tau}, \tag{2.5}$$

where $\sim$ denotes the filtered variables and $\boldsymbol{\tau}$ is the sub-grid scale (SGS) stress tensor, expressed by the fluctuating velocity components u_i' as

$$\boldsymbol{\tau}_{ij} = \rho \left(\overline{u_i' u_j'} - \overline{u_i'}\,\overline{u_j'} \right). \tag{2.6}$$

The eddy-viscosity assumption first introduced by Boussinesq [**?**] is a common way to simplify this tensor, which will be done in this work as well,

$$\boldsymbol{\tau}_{ij} = -2\mu_{SGS} \overline{S}_{ij} + \frac{1}{3} \delta_{ij} \boldsymbol{\tau}_{kk}. \tag{2.7}$$

Notice that the isotropic part of the tensor, $\boldsymbol{\tau}_{kk}$, is not modelled, but will instead be added to the filtered static pressure term $\tilde{p}$. The tensor $\boldsymbol{\tau}_{ij}$ will model the effect of the SGS motions in the flow. There are many ways of modelling this tensor, and for this work the dynamic Smagorinsky-Lilly [**?**] model is used. This is a method developed from the model originally proposed by Smagorinsky [**?**], with some minor modifications,

$$\begin{aligned} \mu_{SGS} &= \rho L_s^2 S, \\ S &= (2\bar{S}_{ij}\bar{S}_{ij})^{1/2}, \\ L_s &= \min(\kappa d, C_s V^{1/3}), \end{aligned} \tag{2.8}$$

where κ is the von Kármán constant and d is the distance to the closest wall. The Smagorinsky constant C_s is calculated dynamically [**?**], V is the cell volume and L_s is the filter width.

3 Computational details

All simulations are performed in Fluent 16.1 and the settings used can be found in subsection 3.1. The least conventional part of the implementation is how the inflow boundary condition is imposed. The velocity profile entering the domain is generated in a previous simulation [**?**]. Figure 3.1 illustrates how this velocity profile is imposed at every timestep as an inflow boundary condition at $x = 1.04$. Along the sides of the channel and the roof symmetric boundary conditions are selected, the floor has no-slip conditions and for the outflow plane the "do nothing" BC is imposed.

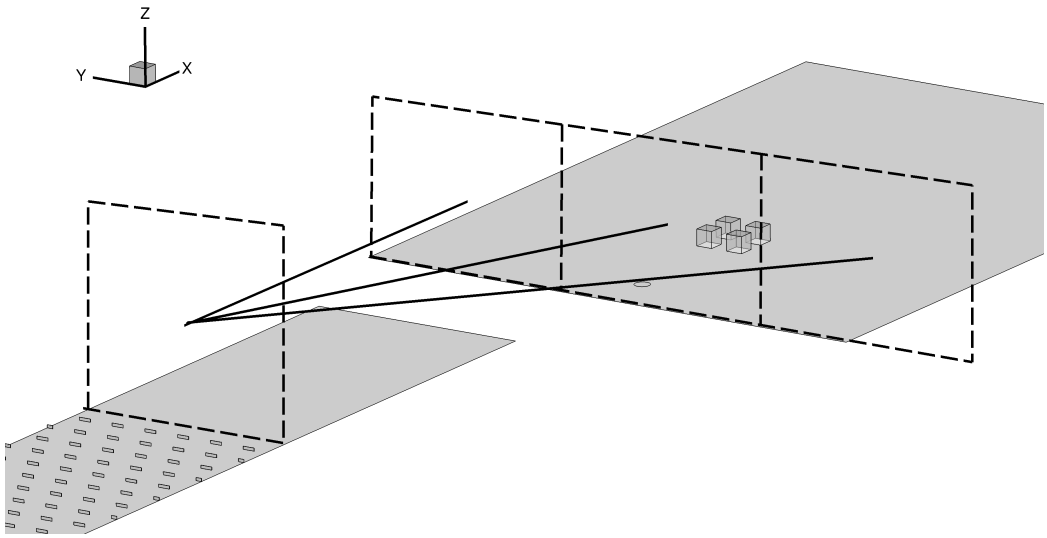

Figure 3.1 The instantaneous velocity field from the previous simulations is inserted as the inflow boundary condition. The small shaded strips represent roughness elements used for creating the boundary layer.

3.1 Implementation in Fluent

The configurations used are briefly stated in the list below. The boldfaced objects are the default settings which will be compared with optional settings.

1. Models
   - Viscous – LES with Smagorinsky SGS model
   - Species Transport using a volume weighted mixing law
2. Materials
   - Defining air and the gas to be released
3. Boundary conditions

- Main inflow: Read from files using several UDFs
- Gas source: Specified mass flow rate $50l/\min$
- Floor and cubes: No slip conditions
- Walls and "sky"Free slip conditions by specifying symmetric stress

4. Solution Methods
   - Pressure-Velocity Coupling Scheme: SIMPLE
   - Gradient: Least Squares Cell Based
   - Pressure: Second Order
   - **Momentum: Bounded Central Differencing**
   - **Air: Second Order Upwind**
   - **Transient Formulation: Bounded Second Order Implicit**

In addition to these standard settings various user defined functions (UDFs) had to be implemented in order to define the input velocity. The solution methods stated in the list above are the default settings for a problem of this type. The discretization for both the momentum equation and the time-development are bounded, which implies the use of some extra parameters to stabilize the solution. A second order implicit discretization in time is given as

$$\frac{\partial \phi}{\partial t} = \frac{3\phi_n - 4\phi_{n-1} + \phi_{n-2}}{2\Delta t}. \tag{3.1}$$

This discretization provides an error term of order $O(\Delta t^2)$. The bounded second order implicit scheme provides a similar discretization with the bounding factors $\beta_{n+1/2}$ and $\beta_{n-1/2}$. How these variables are chosen is not stated explicitly in the Fluent Theory Guide [**?**], but the scheme is given as

$$\frac{\partial \phi}{\partial t} = \frac{(1+\beta_{n+1/2})\phi_n - (1+\beta_{n+1/2}+\beta_{n-1/2})\phi_{n-1} + \beta_{n-1/2}\phi_{n-2}}{\Delta t}. \tag{3.2}$$

Unless the bounding factors are chosen such that they correspond to the discretization in Eq. (3.1) the error term will be of order $O(\Delta t)$. The second order upwind method used for the discretization of the species is a method which adds a diffusion term of order $O(\Delta t)$ and does in this way stabilize the numerical method by reducing the order of accuracy.

By choosing different solution methods one can expect to obtain slightly different run times and solutions. Based on the assumption that Fluent provides stable, diffusive solutions the momentum, air and transport schemes from point 4 in the list above will be exchanged for less diffusive schemes.

3.2 Reading the inflow

The inflow at each timestep is specified componentwise in separate files. For each timestep the file corresponding to the current flow time is read and interpolated to the nodes defining

the inflow in Fluent. It is necessary to take advantage of the built-in `DEFINE` macros Fluent has available, more information about these can be found in the *Fluent UDF manual* [**?**]. The whole process is divided into the following three steps:

1. `DEFINE_INIT(name,*d)` - Defining the interpolation algorithm. Only executed when the flow is initialized.
2. `DEFINE_ADJUST(name,*d)` - Reading the inflow for the current timestep and defining the components of the inflow. Executed at the beginning of each timestep.
3. `DEFINE_PROFILE(name,*t,i)` - Applying the predefined interpolation algorithm and imposing the interpolated values to each node in the inflow.

In order to clarify the approach the general pseudo code is listed below.

```
/* Pseudo code */
define interpolation algorithm  (‘‘DEFINE_INIT’’)
for each timestep T
  Read inflow (‘‘DEFINE_ADJUST’’)
  for each node N and velocity component V on inflow boundary
    V(N) = doInterpolation(N) (‘‘DEFINE_PROFILE’’)
  endfor
  solve
endfor
```

In order to make the interpolation algorithm run efficiently some global variables are stored. These variables will be deleted after a Fluent session and it is necessary to redefine them for every new Fluent session. Note that `DEFINE_INIT` is only executable at the initialization of the flow. There is therefore necessary to initialize the global variables in a different way. This is solved by creating the UDF `DEFINE_ON_DEMAND` which can be executed at any given point and thus initialize the global variables when starting up Fluent in the middle of a simulation.

The interpolation algorithm used for this experiment is an inverse distance interpolation using the four closest points and the inversed distance squared as weights.

3.3 Cases investigated

One of the main objectives of this work is to investigate the Fluent solver and how it manages to simulate gas dispersion. In order to get a good understanding of how the solver works and how the different settings in Fluent affect the solution, several comparison studies were performed. The results are presented in chapter 4, but for clarity the different cases investigated will be stated and motivated here.

1. **Fluent/CDP/Wind tunnel experiment**
   comparing the solution to the data from CDP simulations and wind tunnel experiments

2. **Variation in solver settings**
   Fluent automatically suggests certain solver settings depending on the problem to be solved. To find out if it was possible to make the solver more efficient and accurate, it was experimented with several of the schemes available.
3. **The effect of the SGS model**
   Comparing solutions with and without an SGS model. In the central parts of the domain the mesh is very fine, $\Delta \sim \eta$, where η is the Smagorinsky length scale which is estimated to be approximately 1.5mm in section 4.5. Most of the turbulent structures are resolved and the SGS model should be less important.
4. **Variation in gridsize**
   In order to find out how good the resolution of the mesh needs to be, solutions using $N = 6.200.000, N/2$ and $N/4$ nodes is compared. For the coarser meshes the SGS model will be more important, and a lot of computational time can be saved if the SGS model is found to provide good results. This is also an important test to check that grid convergence is obtained.
5. **Fluent/CDP/Wind tunnel experiment - Heavy gas**
   Comparing the solution to the data from CDP simulations and wind tunnel experiments for heavy gas release. The same volume of gas is inserted to the domain, and the settings of the solver is as similar as possible to the default case for neutral gas. The only changes necessary in Fluent was switching on gravity and adding CO_2 as a species.

4 Results and discussion

This section contains the results from the simulations compared with the experimental wind tunnel measurements and the simulations performed with CDP. In order to generate the results the values in table 4.1 are being written to file from Fluent.

x_i	Coordinates
$u_i u_i$	Turbulent normal stresses
$u_1 u_3$	Reynold's stress component
U_i	Velocity components
C	Concentration of released gas
c	Fluctuating part of the concentration

Table 4.1 Output variables from simulations performed with Fluent.

All values are either predefined in Fluent or easily calculated from the available values. All values are averaged in time, with a sample time specified in each case. All time-averaged variables will be denoted $\overline{var}$, and the variables averaged in both time and space will be denoted $\langle var \rangle$.

4.1 Boundary layer creation

Before any gas can be released the profile of the flow and the boundary layer have to be correctly estimated. The Reynold's stresses and the scaled velocity variable is therefore compared with the wind tunnel experiment. The results in figure 4.1 show good agreement with the experimental data apart from an underestimation of the first normal stress. The sharp peaks right above ground level are due to the roughness elements (see figure 3.1) used to create the boundary layer. One can clearly see from $\langle uu \rangle$ how this stabilizes inside the domain where no roughness elements are present.

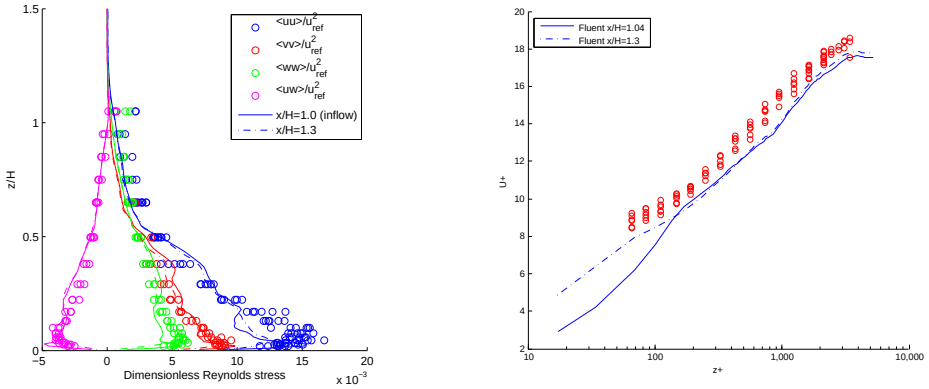

Figure 4.1 *Lateral ($-1 \leq y/H \leq 1$) and time-averaged wind field at the inflow and at $x/H = 1.3$, compared with experimental data. (a) Non-dimensional Reynold's stress plotted against z/H. (b) U^+ plotted against z^+*

The reduction of stress is also visible in 4.1(b). $U^+ = U_{ref}/U^*$ increases slightly in the buffer region ($5 < z^+ < 200$) implying a reduction in the friction velocity estimated by the relation $U^* = \sqrt{-\langle uw \rangle}$. The averaging of uw in this particular estimation of U^* is done in time and within the spatial domain $-1 \leq y/H \leq 1$ and $0 \leq z/H \leq 0.2$.

4.2 Neutrally buoyant gas

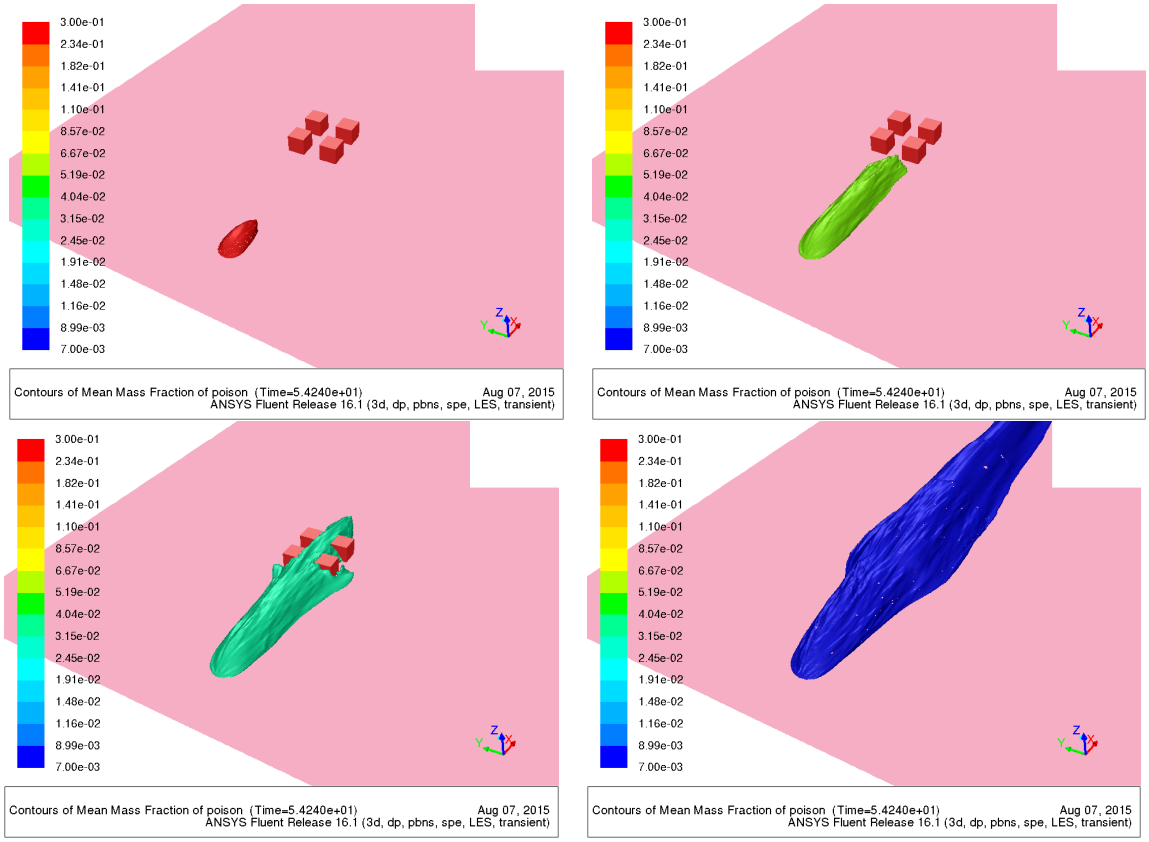

Figure 4.2 *Species concentration plotted on iso-surfaces of time-averaged mass fraction for neutral release. The sample time is* 11.04 *seconds. (a)* $C = 0.3$*, (b)* $C = 0.06$*, (c)* $C = 0.03$*, (d)* $C = 0.007$*.*

In figure 4.2 the mass fraction of the released gas is fixed on the surfaces. The results show a very smooth surface for the different values for the concentration. The plume is slightly shifted towards negative y-direction, this indicates that the sampling time should be even longer.

Figure 4.3 shows the scaled concentration along the dashed lines in figure 1.1. The experimental data and Fluent provide a plume that is slightly shifted towards negative y, whereas CDP has a shift towards positive y. The shift in the experimental data could be explained by some geometrical asymmetry or insufficient time used for measuring. The simulated results are probably not symmetric because the sampling time is not sufficiently long, and since the simulations are invoked at different times the asymmetry is different. Another explanation could be that in Fluent the inflow condition is translated so that it corresponds to a symmetric distribution of roughness elements in the creation of the boundary layer. The simulated width of the plume is consistent with the experimental data, and the maximal concentration is also very similar, but somewhat higher for the measurements downstream in the experimental data.

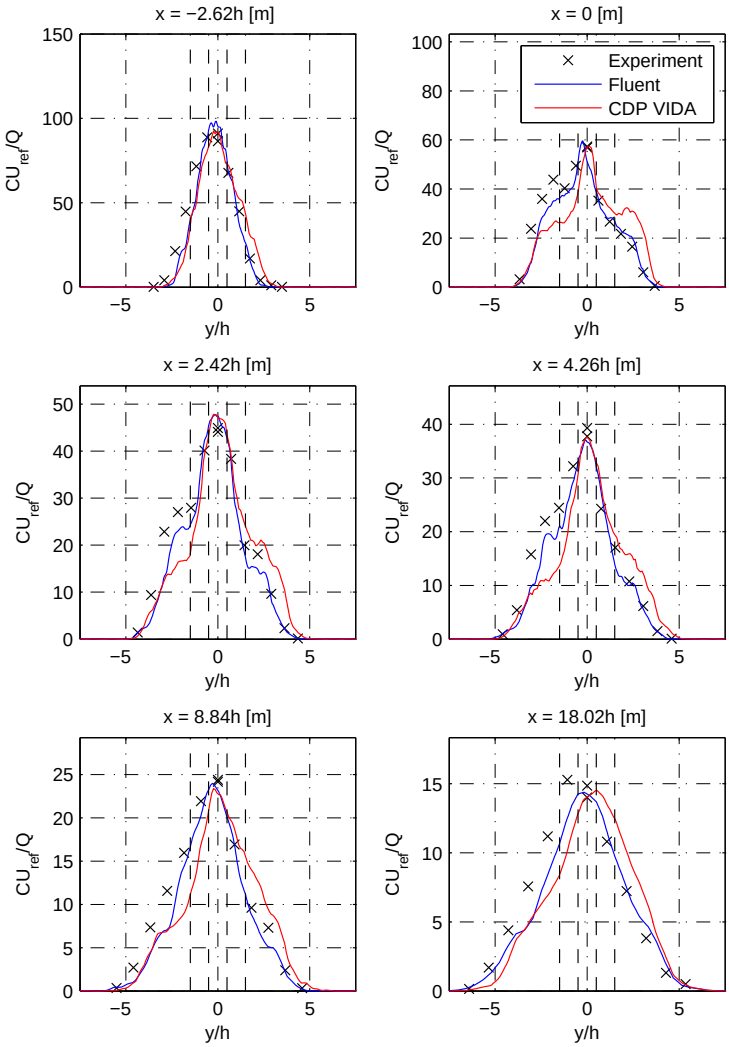

Figure 4.3 *Time-averaged concentration with a sample time of* 12.48 *s at* $z/H = 0.025$ *plotted horizontally and scaled with the free-stream velocity and emission rate. Compared against wind tunnel data. Two dashed lines on either side of the centerline represent the canyon.*

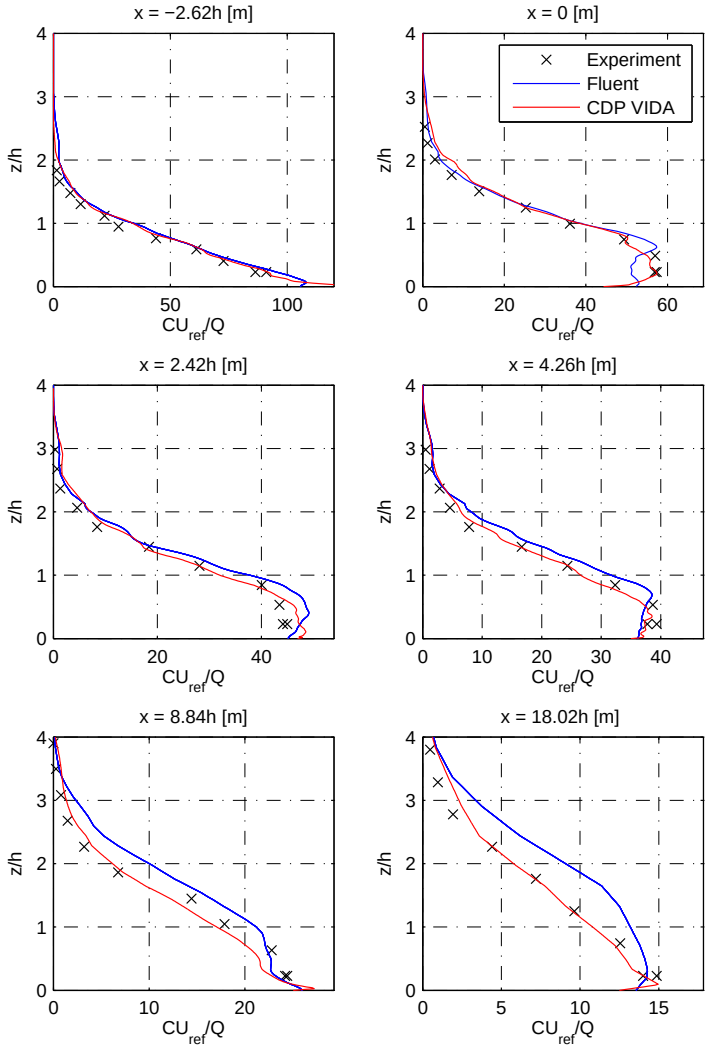

Figure 4.4 *Time-averaged concentration with a sample time of* 12.48 *s at* $y/H = 0$ *plotted vertically and scaled with the free-stream velocity and emission rate. Compared against wind tunnel data.*

The computational results in figure 4.4 capture very well the structure of the concentration in the plane $y = 0$. Further downstream the differences increase, and Fluent has a tendency to overestimate the concentration. Close to the wall the solutions are not identical, but all the data estimates roughly the same peak concentration.

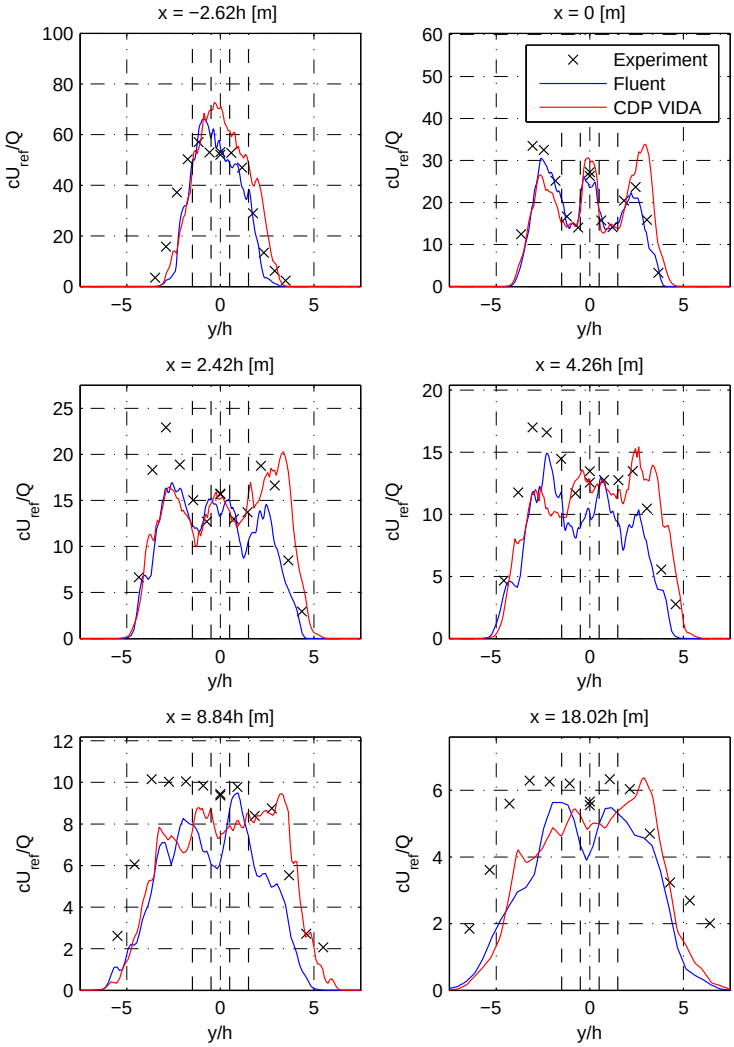

Figure 4.5 *Fluctuating concentration with a sample time of* 12.48 *s at* $z/H = 0.025$ *plotted horizontally and scaled with the free stream velocity and emission rate. Compared against wind tunnel data.*

The fluctuation of the released gas concentration is plotted against the wind tunnel data in figure 4.5. Also in this plot the simulated and experimental data give similar results. One can clearly see how the simulation captures the structures of the fluctuation.

4.3 Performance testing - (removed/do again due to cluster instabilities?)

A small performance test was performed to evaluate Fluents ability to run on multiple cores. In table 4.2 the simulations are not executed on the same interval in time, but the flow and gas-release is well-developed in all 3 cases. This can affect the convergence-rate and thus also the total run-time. The data does however indicate a good speedup for a number of

Cores	Time (h)	Speedup per iteration	Total speedup
40	94,7	1	1
80	51,6	1,9	1,8
120	25,8	2,9	3,7

Table 4.2 Small performance test on 4000 iterations.

cores in this range. For 120 cores the convergence went drastically faster than for the two others, this is what cause the surprisingly good total speedup. A possible explanation could be that the local preconditioner used in the calculation results in a faster convergence when the problem is decomposed to 120 cores.

4.4 Testing of discretization schemes in Fluent

As mentioned in Section 3.1, several of the solution methods have been tested and a small performance-test is done. The discretizations referred to as default are the ones stated in boldface in the list in Section 3.1. The performance data are all relative to the all-default solution given explicitly in the last row. Relative convergence refers to the number of times convergence is reached relative to the all-default case.

	Discretization			Relative performance		
Settings	Time	Momentum	Species	Time per it	Convergence	Total time
1	2nd	Default	Default	0.92	1.2	0.90
2	Default	3rd MUSCL	Default	0.99	1.4	0.83
3	Default	Default	3rd MUSCL	1.01	1.2	0.99
4	2nd	3rd MUSCL	3rd MUSCL	0.93	1.4	0.80
Default	2nd bounded	bounded CD	2nd upwind	1.85 sec	69%	94.7 hours

Table 4.3 Performance test on 4000 timesteps, or 6.4 seconds for different solution method settings in Fluent, all with convergence criteria $5 \cdot 10^{-8}$ on the residual from the continuity equation, and a maximum of 50 iterations per timestep. The simulations are all performed on 40 cores. Convergence is here defined as the number of timesteps which reached convergence divided by the total number of timesteps.

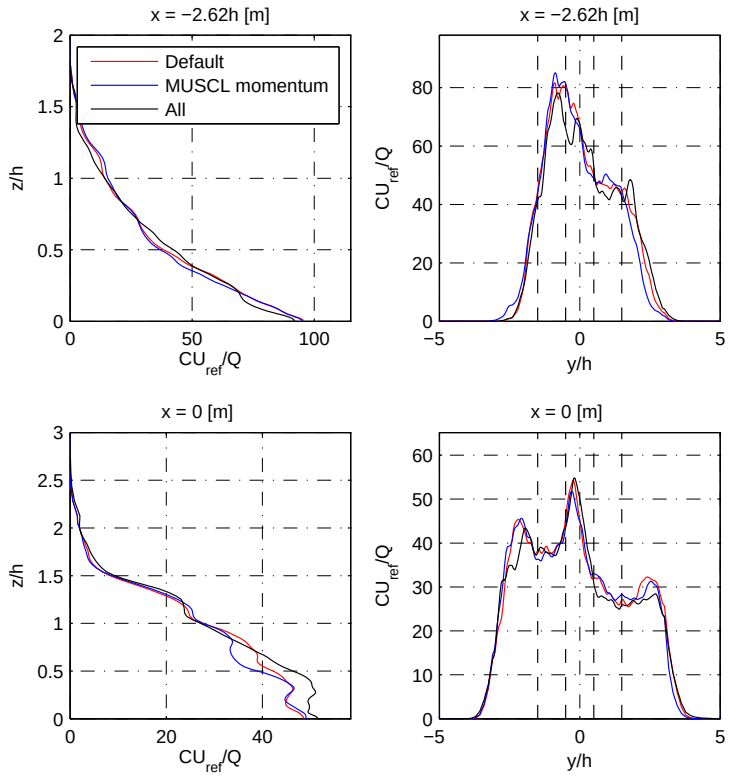

Figure 4.6 Time-averaged concentration with 5.92 *s of sampling at* $z/H = 0.025$ *plotted horizontally and scaled with the free-stream velocity and emission rate.*

The performance results presented in table 4.3 reveal that the performance is not affected in any significant way by the scheme chosen for the species. This is as expected since the simulation is done with a neutrally buoyant gas which will have very little impact on the flow. By changing the time discretization the convergence rate increases slightly, while the calculation time drops by almost 10 %. With the 3rd order MUSCL scheme for momentum, convergence is increased by 70 %, and with a small change in the computational time at iteration level the total runtime is decreased by 17 %. The data in table 4.3 will depend on the convergence criteria and the maximum number of iterations allowed, but it gives an indication of how the different solver settings compare. The change which has the most significant impact is the momentum discretization, in figure 4.6 the results after 5.92 seconds is presented. The shape of the plume is very similar in all cases, the width is identical and the height is also very similar. There are however small deviations that could potentially make the plumes even more distinct over time, in order to better understand the effect of the schemes the simulation was done over a larger time-interval.

Since discretization settings nr. 4 in table 4.3 (from now on referred to as the tuned case) performed best this was a natural choice for a longer comparison against the default case. The results are presented in figure 4.7. After a longer sampling time the effects of the default bounded schemes recommended by Fluent become clearer. Although the results have similarities with respect to the shape of the plume the tuned case consistently underestimates the concentration.

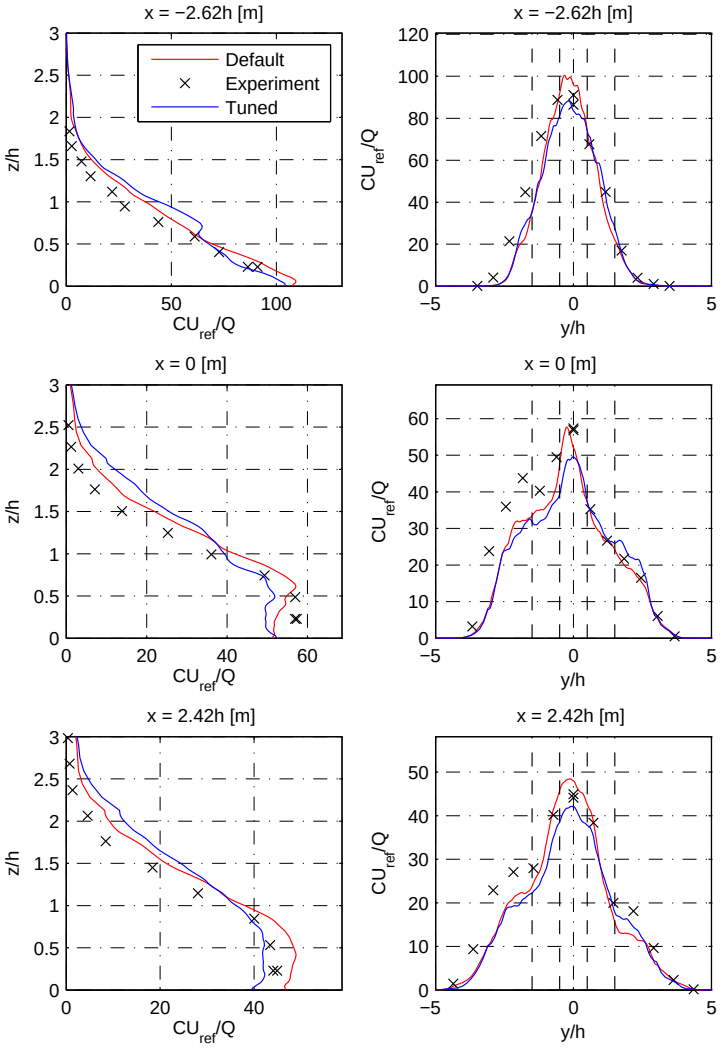

Figure 4.7 *Time-averaged concentration with 12.48 s of sampling at $y = 0$ to the left and $z/H = 0.025$ to the right. Both plots are scaled with the free-stream velocity and the emission rate.*

4.5 The effect of the subgrid-scale model

Performing an LES implies resolving the larger turbulent structures and modelling the smaller ones using an SGS model. In this case the modelling is done using the Smagorinsky-Lilly SGS model. By using a sufficiently fine grid, the effect of the unresolved structures will have very little influence on the rest of the flow. The viscous length scale is the scale that is appropriate to use in order to resolve all structures. An estimation of this scale is given as

$$\delta_v = \sqrt{\frac{\nu}{\mathcal{S}}} \approx 1.5\text{mm} \tag{4.1}$$

ν is here the efficient kinematic viscosity and $\mathcal{S} = |S_{ij}|$ is the magnitude of the strain-rate tensor. The estimated value is calculated along the dotted measurement lines in figure 1.1.

The grid size $\Delta = V_{cell}^{1/3}$, which in this case is the same as the filter size used in this work, is of the same order as δ_v. The fact that $\Delta \sim \delta_v$ means that the SGS model might not be as important since most of the energy spectrum is resolved. A comparison was done between dynamic Smagorinsky and no SGS model, the results are plotted in figure 4.8. The plots clearly show that even without the SGS model Fluent is able to capture the shape of the plume. It is worth to mention that the sampling time is not sufficiently long for the plume to be stabilized, so it is hard to determine whether the simulation without an SGS model will be able to predict a plume that is similar to the measured data from the wind tunnel experiment. But the effect of the SGS model is indisputable, especially the horizontal plots illustrates how it reduces the asymmetry of the plume.

By comparing the performance as done to the tests in table 4.3 one finds that by omitting the SGS model the computational time per iteration decreases with less than 5% and that convergence is about 10% slower. The resulting effect is that the total computational time does not change.

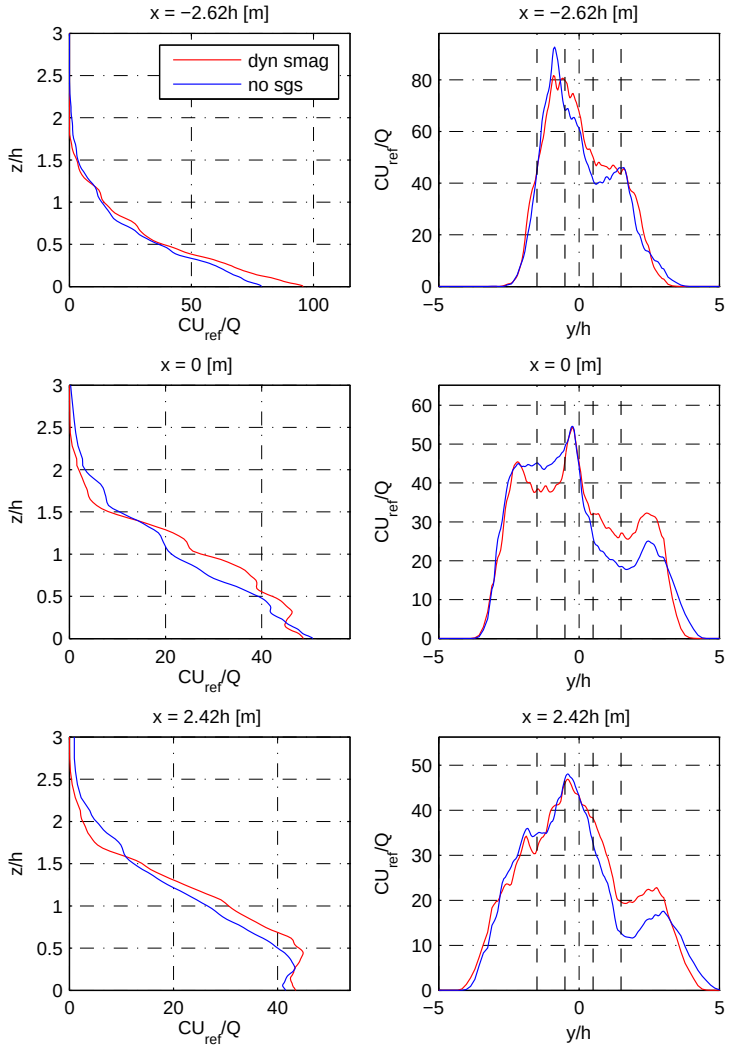

Figure 4.8 *Dynamic Smagorinsky compared with no SGS model. Time-averaged concentration with 5.92 s of sampling at $z/H = 0.025$ plotted horizontally and scaled with the free-stream velocity and emission rate.*

4.6 Grid size

By reducing the number of grid cells, the result should yield a less accurate solution. In figure 4.9 three distinct meshes are compared. The difference between the solutions are not critical, they all estimate a plume similar to the experimental one. With coarser mesh Fluent does however consistently estimate a lower peak, something that would suggest a poorer approximation of the conserving laws of passive scalars. The semi-coarse mesh is

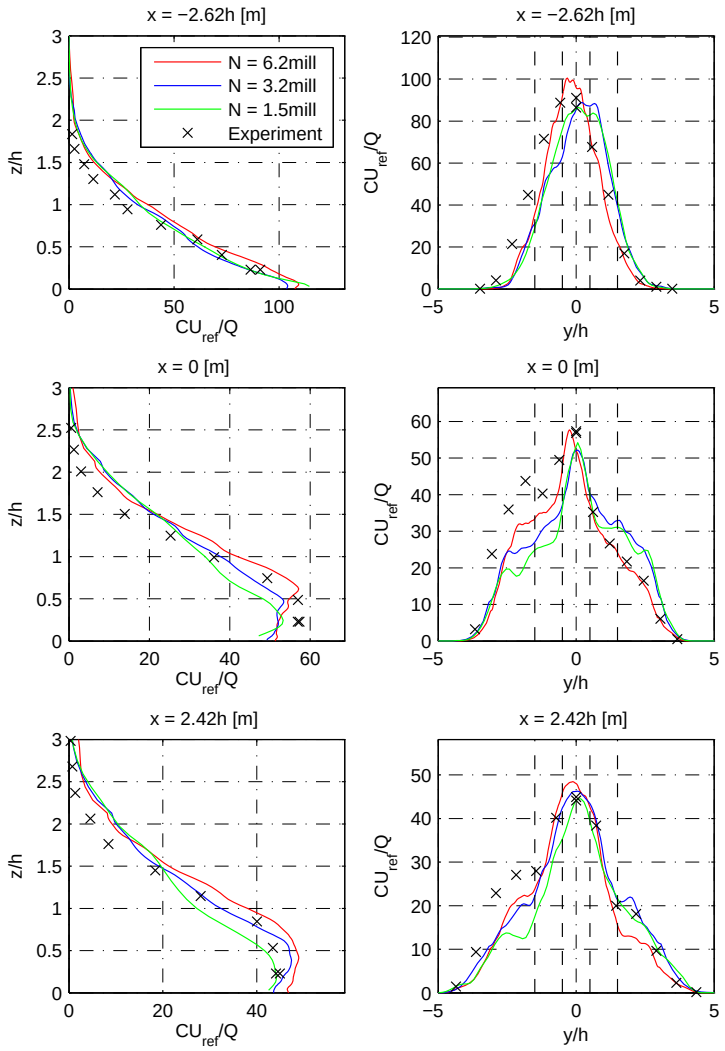

Figure 4.9 *Time-averaged concentration with* 12.48 *s of sampling at* $y/H = 0$ *to the left and* $z/H = 0.025$ *to the right. All concentrations are scaled with the free-stream velocity and the emission rate.*

created by lowering the resolution along the edges, but the resolution-levels are maintained. More specifically Δx and Δy from table 1.1 is multiplied by a factor of 1.4 while Δz remains

unchanged. The coarsest mesh is identical to the semi-coarse mesh but without the finest refinement layer, in other words $R1 = R2$.

5 Heavy gas

A passive scalar does not have any effect on the velocity field so the computational challenges are similar to the ones encountered when simulating a common flow problem. When a heavy gas is released into a flow it has to be treated as an active scalar and it will affect the initial velocity field. This is a computationally more demanding problem, but nevertheless a realistic case and it is important to verify that our computational software is able to provide good estimates. The measurments are done in the same fashion as for neutral gas, but at slightly different positions downstream.

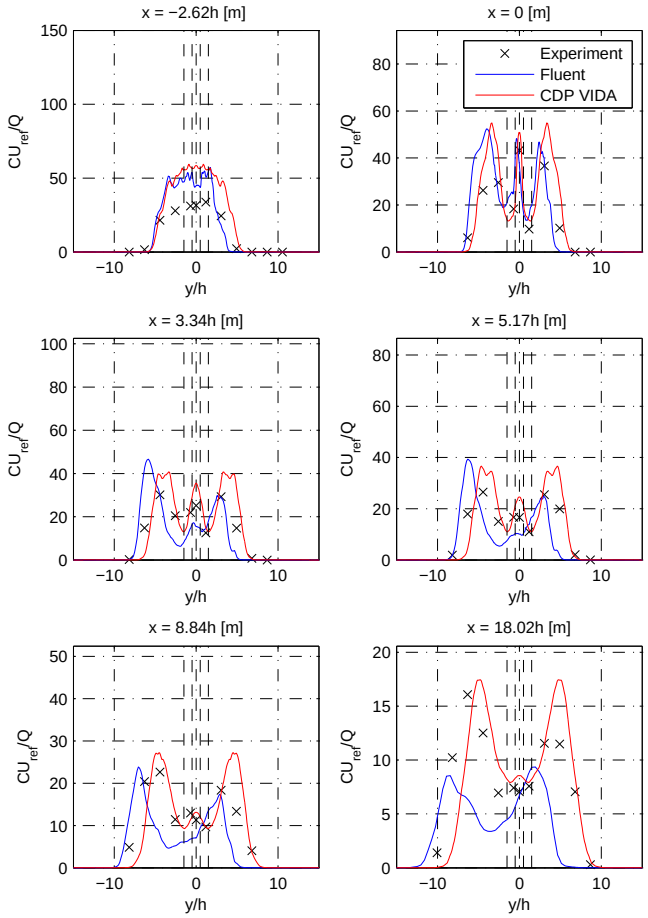

Figure 5.1 Time-averaged concentration with 28.8 s of sampling at $z/H = 0.025$. All concentrations are scaled with the free-stream velocity and the emission rate.

Along the first two measurment lines in figure 5.1 Fluent and CDP provides very similar results. And the measurement at $x = 0$ shows decent similarity to the experimental data as well. Further downstream Fluent fails to predict the middle peak which is present in both the experiment and the simulations done in CDP. The concentration is also consistently underestimated.

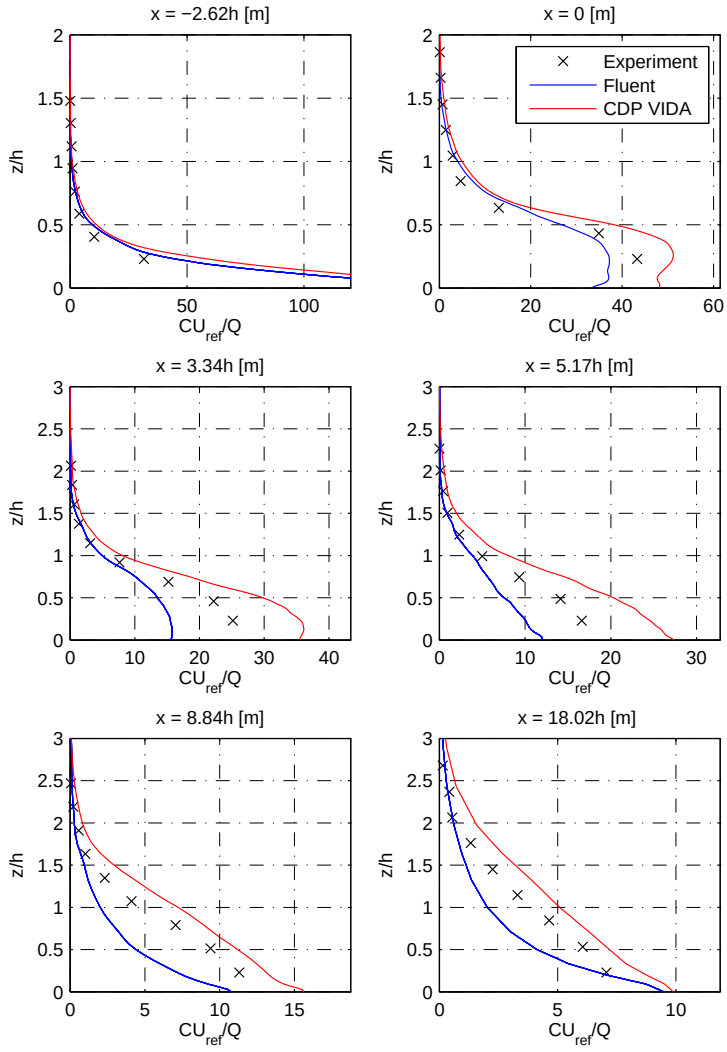

Figure 5.2 Time-averaged concentration with 28.8 s of sampling at $y/H = 0$. All concentrations are scaled with the free-stream velocity and the emission rate.

From the data along the vertical lines plotted in figure 5.2 Fluent consistently underestimates the concentration close to the wall. The slope of the profile is also very different from CDP and the experimental data.

6 Conclusion

Fluent is compared with simulations done in CDP and experimental data. All simulations are done using LES. It is also experimented with different discretization schemes and the effect of an SGS model in Fluent. The test case is release of a neutrally buoyant gas in a turbulent boundary layer over an urban street canyon. The data used for comparison is time-averaged concentration along certain horizontal and vertical lines. The width and height of the plume computed in Fluent is consistent with both experimental data and simulations in CDP. There are some disagreements in the data measured close to the wall and far downstream.

The importance of the SGS model is also confirmed, although even without an SGS-model Fluent is able to predict a similar height and width of the plume. The shape of the plume is however somewhat distorted when omitting the SGS model. The increased accuracy achieved by increasing the degrees of freedom confirms that the model is well-functioning.

Different choices of discretization schemes can save up to 20% of the computational time, but fail to provide more accurate results. The results from this case suggests that the default settings in Fluent provide the most accurate result.