AUTOMATIC GRID GENERATION FROM A NUMERICAL PICTURE FOR TRANSIENT FLOW SIMULATION OVER A CAR SHAPE OBSTACLE

Tonino Sophy[(a)], Julien Jouanguy[(b)], Luis Le Moyne[(c)]

[(a)] DRIVE – ISAT EA 1859, 49, rue Mademoiselle Bourgeois, 58000 NEVERS, FRANCE.

[(a)] tsophy@u-bourgogne.fr , [(b)] julien.jouanguy@u-bourgogne.fr , [(c)] luis.le-moyne@u-bourgogne.fr

ABSTRACT

Computer assists numerization of a domain, requires several engineers or scientists during considerable time. Thus, meshing automatization process has been developed using heavy devices like LASER metrology. It can sometimes be more convenient to use simple devices.

Image processing field, reveals many works concerning object detection. Applications concern medical field, automotive, face detection or national defense.

This paper aims proposing a simple, but accurate enough, tool to generate 2D domain meshing from a numerical picture that can be used with a transient Finite-Volume CFD code. A car shape object is chosen. From the original picture, edge detection and threshold techniques are applied and then, the object contour points' gird is provided.

Another process is applied to generate a refined Finite-Volume mesh compatible with Gerris Flow solver. Transient simulations are conducted at different Reynolds numbers. Results, in terms of pressure fields or velocity evolutions, are shown. Von-Karman alley flow is detected.

Keywords: Flow simulation, Object Recognition, Complex shape, Image processing, Mesh generation, Drag coefficient.

1. INTRODUCTION

Numerical simulation is widely used in advanced technology studies especially in engineering field. Indeed, it is necessary to use adapted numerical method to produce convincing simulations of physical phenomenom such as fluid flow or thermal diffusion. Even if direct experimentations are essential, the observation of physical phenomenon is more convenient using numerical simulation.

Numerization of the considered domain (often called meshing), that is usually achieved with computer assisted conception software, can require one or more engineer or scientist during a considerable amount of time. Thus, automatization of meshing process has been developed using heavy devices like LASER metrology. Nevertheless, it can be more convenient to use a simple device, like a commercial digital camera, to run simple numerical tests.

Image processing scientific field, reveals many works done concerning object detection or reconstruction. Main applications concern medical field, automotive comfort or security, face detection (Viola 2001, Li 2002, Sochman 2004) or national defense department. Muñoz-Salinas et al. used image processing to detect and track people with multiple stereo cameras (Muñoz-Salinas 2009). Cao et al. used image treatment to reconstruct surface using bivariate simplex splines on Delaunay configurations (Cao 2009).

The aim of this paper is to propose a simple, but accurate enough, tool to generate a 2D domain meshing from a numerical picture that can be used with a transient FV (Finite Volume) CFD code. A car shape object is chosen. A description of the grid generation process, from the original picture to the object contour points' gird, is first provided. Then, the process to load these contour points and to generate a refined Finite Volume mesh compatible with the Gerris Flow solver is rapidly described. The tool is validated using a car shape object. The known Von Karman alley phenomenon can be observed. Thus, transient simulations are conducted at different Reynolds numbers. Results, in terms of pressure fields or velocity versus time evolutions, are shown. This type of results can allow evaluating the drag or lift coefficient of the car or the fundamental vortexes' drop frequency. The influence of the Reynolds number on the vortexes' drop frequency is depicted.

2. METHOD DESCRIPTION

The main purpose of the proposed tool is using a simple commercial digital camera picture to build a 2D mesh in order to run a numerical simulation. When an image is acquired by a camera for another vision system, it is rarely directly ready for use. Thus, a first image processing should be applied for noise reduction for instance. Then, the real image processing can take place to obtain the points cloud representing the picture. At last, this points cloud has to be modified to be load in our numerical simulation software. The obtained mesh is so ready for the simulations.

2.1. Image processing

In this section we show different used methods for image treatment. A brief description will be given for these methods. Our code development is based on C++

language using an Open source Computer Vision library (OpenCv) (Bradski 2008). To achieve the points grid generation, the captured image has to be smoothed by filtering. This is necessary for edge detection then pixel classification.

2.1.1. Mathematical tool

To perform image preprocessing and processing several mathematical tools are required like linear discrete convolution or derivative filters.

- Linear convolution

Consider an image I[x, y] and a two-dimensional filter h [x, y]. The 2D discrete convolution sum is then given by equation 1.

$$g[x,y]=\sum_{k=-\infty}^{\infty}\sum_{l=-\infty}^{\infty}I[k,l]h[x-k,y-l] \quad (1)$$

Then g [x, y] is the new obtained image after convolution.

- Derivative filter

In practice, to work with discrete gradients it is necessary to approach them. It is usual to see a Finite Difference scheme calculated by a convolution with simples kernels as an approximation of directional derivative. For example, the approximation $\frac{\partial I}{\partial x}$ of the derivative of a continuous signal I(x) is obtained by convolution with the simple kernel [0 -1 1]:

$$\frac{\partial I}{\partial x}\approx\frac{\Delta I}{\Delta x}=\frac{I(x+\Delta x)-I(x)}{\Delta x} \quad (2)$$

If the signal I is an image we have two derivatives at each point (pixel). The vertical and horizontal derivatives are respectively Iy and Ix. Then, the gradient image is defined as the sum of a two components

$$\Delta I=|I_x|+|I_y| \quad (3)$$

ΔI gives an indication of the intensity of the gradient in the current pixel.

2.1.2. Image Preprocessing

When an image is acquired by a camera or other imaging system, often the vision system for which it is intended is unable to use it directly. The image may be corrupted by random variations in intensity, variation in illumination, or poor contrast that must be dealt with in the early stages of vision processing. So, good image smoothing should be able to deal with different types of noise. In this paper, some image smoothing filters are used.

In this paper, non-linear and linear filters are the two types of smoothing methods used.

- Linear filter

The Gaussian filter is an example of linear filter.

It is used as a linear filter. Its operator is given by equation 4.

$$h_{Gauss}(x,y)=\frac{1}{2\pi\sigma^2}e^{\frac{x^2+y^2}{2\sigma^2}} \quad (4)$$

The kernel of a Gaussian filter with a standard deviation σ = 1.4 is expressed in equation 5.

$$h_{Gauss}=\frac{1}{159}\begin{bmatrix}2&4&5&4&2\\4&9&12&9&4\\5&12&15&12&5\\4&9&12&9&4\\2&4&5&4&2\end{bmatrix} \quad (5)$$

The obtained filter hGauss[x,y] is applied to the image I [x, y] in term of h [x, y] in the Eq.(1).

- Non-linear filter

The median filter is an example of a nonlinear digital filtering technique, often used to remove noise. Instead of simply replacing the pixel value with the mean of neighboring pixel values, it takes the median of those values after they have been sorted. The median is calculated by first sorting all the pixel values from the surrounding neighborhood into numerical order and then replacing the pixel being considered with the middle pixel value.

For example, let's consider a matrix M as expressed Eq. 6.

$$M=\begin{pmatrix}124&126&127\\120&150&125\\115&119&123\end{pmatrix} \quad (6)$$

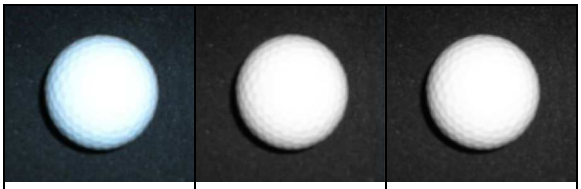
Considering the pixel M(3,3) to be filtered, all the 9 neighbors of this pixel (including the pixel itself) are sorted in a table.

1	2	3	4	**5**	6	7	8	9
115	119	120	123	**124**	125	126	127	150

So the median value, corresponding to the index 5 of the table, replace the value of the M(3,3) pixel and the obtained matrix M_{Median} is expressed in equation 7.

$$M_{Median}=\begin{pmatrix}124&126&127\\120&\mathbf{124}&125\\115&119&123\end{pmatrix} \quad (7)$$

Figure 1 shows an example of original image and the results obtained with a Gaussian linear filter and a Median non-linear filter which is very effective in removing salt and pepper and impulse noise while retaining image details.



(a) Original image | (b) Gaussian filter | (c) Median filter

Figure 1: Smoothing filters

2.1.3. Edge detection

As seen before the contour points of the object are needed by the CFD code. Edge detection is a fundamental tool used in most image processing applications to obtain information from the frames as a precursor step to feature extraction and object segmentation. This process not only detects boundaries between objects and the background in the image, but also the outlines within the object. To detect edges, many operators such as Sobel operator or Canny detector can be applied. The OpenCv library gives an important function that can detect contours. This function is called cvFindContours (Bradski 2008).

During our work many detection operators like Sobel or Canny have been tested. The OpenCv CvFindContour function and the Hough operator emerge from the list (figure 2).

- Sobel operator:

It is a discrete differentiation operator computing an approximation of the opposite of the gradient of the image intensity function. The Sobel operator measures two components. The vertical edge component is calculated with kernel Sy and the horizontal edge component is calculated with kernel Sx. The intensity of the gradient at the pixel is then calculated with the derivative filter. These two kernels are calculated using a simple convolution of [−1 0 1] and $\begin{bmatrix} -1 \\ 0 \\ 1 \end{bmatrix}$ with a smooth filter [1 2 1] (or its transpose):

$$S_x = \begin{pmatrix} -1 & 0 & +1 \\ -2 & 0 & +2 \\ -1 & 0 & +1 \end{pmatrix} \quad S_y = \begin{pmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ +1 & +2 & +1 \end{pmatrix} \quad (8)$$

- Canny operator:

Canny's aim is to find the optimal edge detection algorithm (Canny 1986) based on three criteria:

– Good detection: the algorithm should mark as many real edges in the image as possible

– Good localization: edges marked should be as close as possible to the edge in the real image

– Minimal response: a given edge in the image should only be marked once, and where possible, image noise should not create false edges.

The canny algorithm is built in 5 steps:

– Smoothing: Blurring of the image to remove noise

– Finding gradients: The edges should be marked where the gradients of the image has large magnitudes

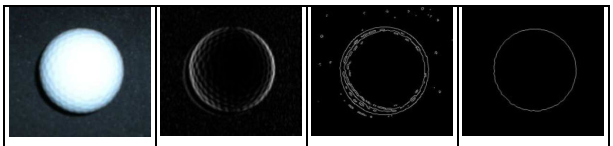
– Maximum suppression: Only local maxima should be marked as edges

– Double thresholding: Potential edges are determined by thresholding

– Edge tracking by hysteresis: Final edges are determined by suppressing all edges that are not connected to a very certain (strong) edge

- cvFindContours:

The function cvFindContours retrieves contours from the binary image and returns the number of retrieved contours. The pointer firstContour is filled by the function. It will contain pointer to the first most outer contour or NULL if no contours is detected (if the image is completely black). Other contours may be reached from firstContour using h_next and v_next links.

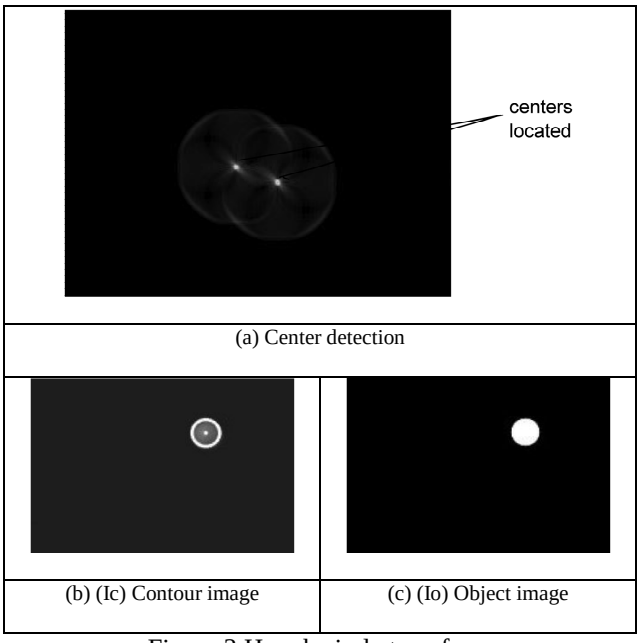


(a) Original image | (b) Sobel detector | (c) Canny detector | (d) Find Contour

Figure 2: Edge detectors

- Hough transform:

The Hough transform is a feature extraction technique used in image analysis, computer vision and digital image processing (Shapiro 2001). It is a method to reconstruct lines, circles, or other simple shape in an image. In our work, as the validation object has a circular shape, we used Hough to reconstruct the circles contour's (Kimme 1975). The Hough circle reconstruction technique highlights in the image the potential centers of a r radius circle (figure 3-a). The center being detected, one can reconstruct the circle's contour (figure 3-b) or the entire object (figure 3-c).



(a) Center detection

(b) (Ic) Contour image

(c) (Io) Object image

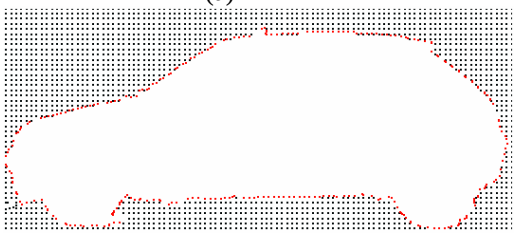
Figure 3 Hough circle transform

2.1.4. Car edge detection:

The initial picture (Figure 4a) used for this study is a 500×222 pixel picture acquired with a Sony DSC-W510 camera. A 72 ppp low resolution have been chosen. The exposition time is 1/250 second.

For the case of the car shape obstacle, the cvFindContours function is used to detect the contour of the car after an accurate pre-processing (regulating the brightness, contrast and Gamma of the picture). Then a contour points grid can be built (Figure 4b) using a threshold level technique.

(a)



(b)

Figure 4: (a): Complex shaped object, (b): Automatic generated grid.

2.2. CFD code mesh import

The Finitie Volume CFD code chosen is the Open Source Gerris Flow solver. This software is licensed under the Free Software Foundation's GPL and is written is C language. As Gerris only deals with 3D calculation, the 2D mesh have to be extruded limited to one cell (intersecting z = 0).

2.2.1. Grid transfer

Our process requires exportation in STL format which is convertible in Gerris object format. CATIA software has been use for the conversion of the text coordinates file to an STL format file. One has to pay attention to the dimensions of the object (aspect ratio) to respect the non dimensional number further.

Finally, the STL format file has to be converted in a Gerris format file (GFS format) using an internal Gerris command. An option (“revert”) can be added to this command to swap the localization of the fluid in or out of the object.

2.2.2. Meshing

Unlike most flow solvers which uses structured or unstructured meshes, Gerris implements a deal between both types by using a tree data structure. For this study, a quadtree data structure has been used to mesh our 2D case. First, 4 quadric volumes of dimension 1 are defined. Then each volume is divided in four new smaller quadrics and the process start again. The volume division process continues until a previously fixed dimension is reached. User has just to precise the number of “level” nlevel to be applied. The volumes size will be $\frac{1}{2^{n_{level}}}$.

Gerris also allows local refinement of the mesh using a wall vicinity technique. In this case, a new level number (nrefine) is required. Near the object, a successive volume division will be applied layer by layer. At the end, the layer closest to the object will have a dimension of $\frac{1}{2^{n_{refine}}}$.

Figure 5 shows an example of refined mesh used in the incompressible 2D Gerris Navier-Stokes’ solver.

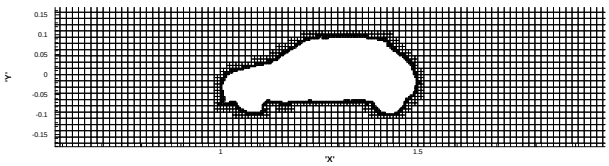


Figure 5: Example of quadric refined mesh representing the car shape obstacle

3. RESULTS

The two dimensional incompressible Navier-Stokes equations are directly solved with Gerris software. All data are adimensionned so that simulations results are only function of dimensionless Reynolds number.

The objective of this preliminary work is to examine whether the method is able to represent the flow over the car in a 2D configuration. Simulation will thus be ran for several intermediate Reynolds numbers ($250 < Re < 500$), for which transient regime is observed without turbulence. The expected

phenomenon is the development, in such a configuration, of a vortex drop behind the car called Von Karman Street which appears behind several geometrical forms with 2D circular or squared sections (Lübcke 2001), (Muddada 2010).

3.1. Simulations configuration

At the inlet, a velocity-step profile is imposed with a single component in the x-direction and convective outflow boundary condition is set to ensure the flow rate conservation. Lateral zero gradient boundaries conditions are also imposed and we checked that their distance from the car is enough to neglect their effect into simulation domain.

Different kind of output data are available such as the pressure field in figure 6 , streamlines on figure 7 or vorticity module on figure 8. These results are obtained from simulations carried out at Re=250 with cells of size 0.0156 (n_{level}=6). The qualitative observations indicate that the transient regime is obtained especially from figure 8 where the vorticity indicate presence of contra-propagative vortexes. On Figure 7, it is also possible to see a recirculation zone behind the car object which is a characteristic of the Von Karman street. For a quantitative study of this regime (detachment frequency and Strouhalt number), it is important to be sure that the grid convergence is reached.

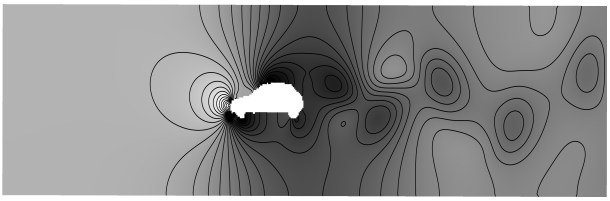


Figure 6: Pressure field for dimensionless time t = 10 and Re=250 . The maximum value in black is 0.7 Pa and the minimum value is -1.6 Pa

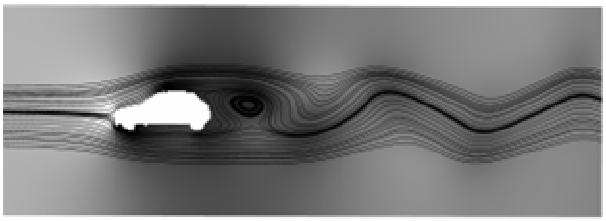


Figure 7: Flow Streamlines at dimensionless time t=10 and Re=250

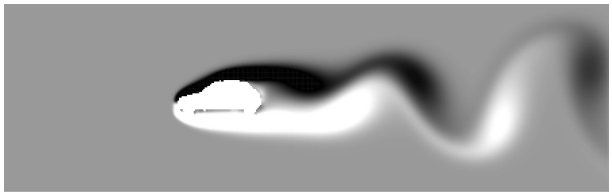


Figure 8: Vorticity module at dimensionless time t=10 and Re=250. Maximum value (+10) is in black and minimum value (-10) is in whit

3.2. Grid convergence

While solving the Navier-Stokes equations, the chosen option is to use a regular grid without distinguishing flow region. The several grid chosen are defined according to the resolution given by the nlevel value and the near object refinement is always given by nrefine=8. The name of the tested grid and their crrespondance to resolution are given in the following table.

Table 1: Different used grid

name	grid1	grid2	grid3	grid4
n_{level}	5	6	7	8

To evaluate the spatial grid convergence we plotted the time evolution of components of the velocity field at position (x=0.125, y=0) at Re=250.

The dimensionless frequency of the signal was also computed from periodicity of the signal after regime stabilization (5<t<10). The convergence is reached when the frequency became independent of the resolution. An example of visualizations is presented on figure 9. On this figure, it is obvious that the grid is very important as it can change the frequency of vortex detachment near the vehicle and also the amplitude of the signal. This can be explained by the fact that Von Karman street is created by boundary layer detachment on the car object. The optimal resolution is thus the one which solve the boundary layer thickness and the whole development of vortexes. The table 1 shows computed frequency for different grids.

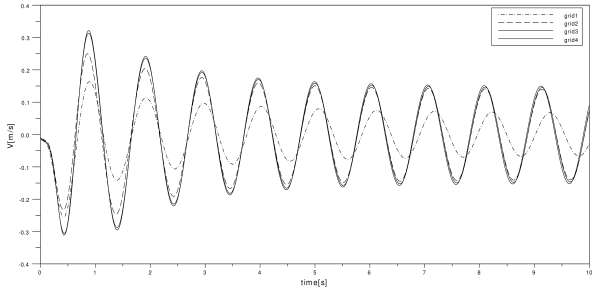


Figure 9: Time evolution of vertical component of velocity V at point (x=0.125,y=0) and Re=250 for several grid resolutions.

Table 2: Frequency of vortex detachment for the different grid tested

Grid n°	1	2	3	4
frequency	1.13	1.16	1.16	1.16

The convergence based on frequency is obtained for grid 2. However, on figure 9 it can be observed that the convergence for amplitude is only reached for grid 3, after the dimensionless time t=3 which is necessary to get a stabilized regime. For the other parts of this study, we will use the same approach to ensure the spatial convergence, especially when the Reynolds

number is increased which may change the boundary layer thickness, and thus the resolution needed for convergence.

3.3. Reynolds number study

Another specific result observed in literature for a transient flow regime behind an object is the relation between Reynolds number and vortex detachment frequency. We thus inquired about this relation with the same procedure as before. The signal of vertical component of velocity in time at point (x=0.125,y=0) is thus analyzed and plotted on figure 10.

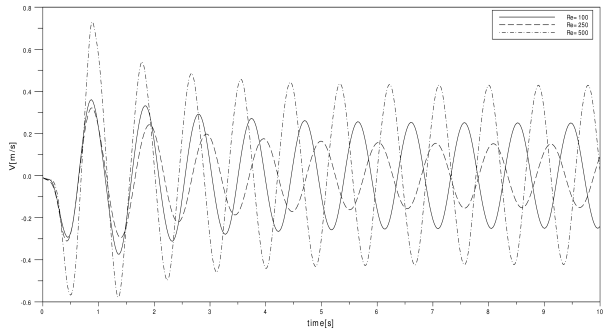


Figure 10 Time evolution of vertical component of velocity V at point (x=0.125,y=0) for several Reynolds number.

The frequency Reynolds dependence can be qualitatively observed. The trends obtained by studies of flow behind other object are also obtained in this configuration, i.e. augmentation of frequency with Reynolds number. The value of the frequencies of different Reynolds numbers can be found in table 2.

Table 3: Frequency of vortex detachment for several Reynolds number

Reynolds number	250	350	500
frequency	1.15	1.22	1.28

3.4. Potential studies with the method

The results of simulation show that this approach, coupled with a CFD code can predict qualitatively the flow comportment behind a vehicle. As the method we used for the flow solver is Direct numerical simulation, the turbulent cases can not be simulated, as a third dimension is needed to get a solution in accordance with experiments. Several option are thus available:

- Extend the method to 3D object description
- Use other CFD aprroach like RANS or URANS with this configuration.

The advantage of this method is that lots of output variables are available and allow computing classical coefficients and forces used to evaluate the aerodynamics of vehicle. For example, Pressure and viscous forces can be obtained and influence of several parameters such as car geometry and Reynolds number can be inquired. An example of such output is presented on figure 11 with evolution of pressure forces that acts on the vehicle. We can see the periodic evolution of these forces in time.

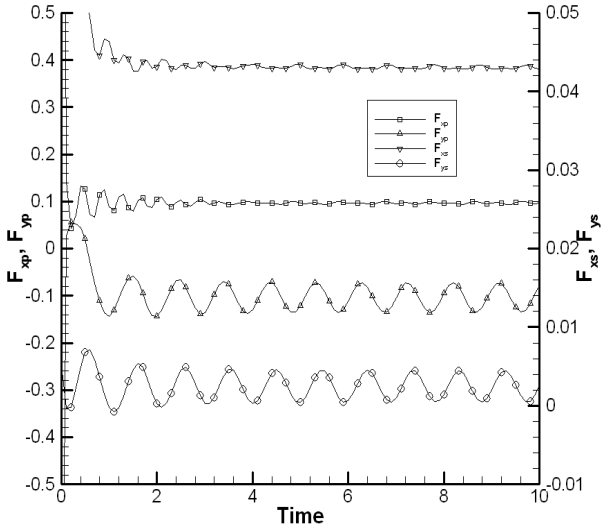


Figure 11 Time evolution of pressure and shear forces on the vehicle

To see if the obtained mesh is robust, 2D high Reynold number simulation is conducted. The purpose is then so see if the Finite Volume mesh leads to convergence of the simulation. The Reynolds number is then fixed to Re = 2.10^5. All other simulation's parameters are kept unchanged. The grid2 is the used.

Figure 12 and 13 respectively show the pressure field at time t = 5 and the evolutions of the components of the velocity at the point (x,y) = (1.625,0) located behind the car shape object for times up to t = 10.

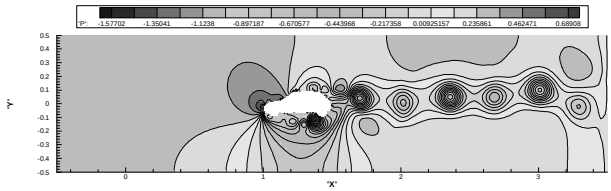


Figure 12: Pressure field for dimensionless time t = 5

Figure 12 reveals that the automatic meshing method is able to generate a mesh compatible with high Reynolds numbers as a converged result is obtained. One can clearly see the iso-pressures corresponding to the vortex drop.

Figure 13 shows that a periodic flow is clearly reached and a dimensionless frequency of 1.9 is found.

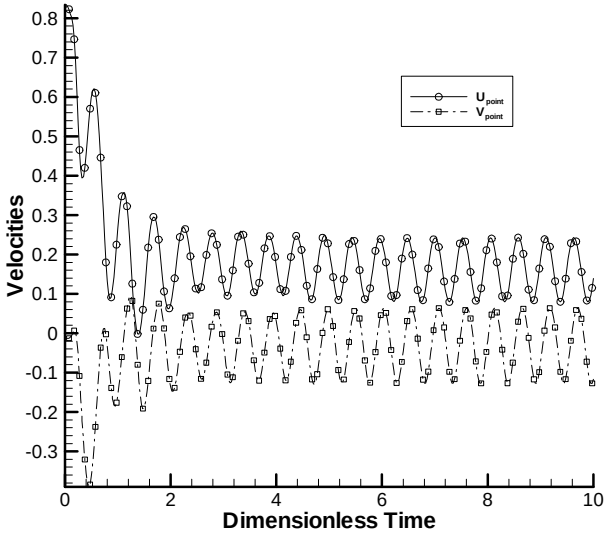


Figure 13: Evolution of U and V velocity component at the point (x , y) = (1.625 , 0)

4. CONCLUSIONS

A light and easy to use automatic grid generation tool as been developed. The idea is to build from a numerical picture, took with a simple commercial digital camera, a realistic 2D numerical simulation mesh. It seems to construct suitable grids for complex shaped object as this work used a car picture as initial picture. The obtained point clouds have been processed to obtain a Finite Volume mesh compatible with Open Source 2D incompressible Navier-Stokes flow solver on transient forced convection problems. Results are in agreement with the physical phenomena as, for several Reynolds numbers, the well known Von Karman Street flow is obtained.

This method potentially allows the study of the optimization of the vehicle flow lift and drag coefficients as the evolutions of the pressure and shear forces are obtained.

Finally, it has been shown that the 2D realistic mesh obtained, does not lead to divergence when high Reynolds numbers are involved.

REFERENCES

Bradski, G.-R., Kaebler, A., 2008. Learning OpenCV,Computer. *Vision with OpenCV Libarary*, O′Reilly Media, Inc., 234-244.

Canny, J., 1986. A Computational Approach To Edge Detection. *IEEE, Trans. Pattern Analysis and Machine Intelligence*, 8 (6), pp. 679-698.

Cao, J., Li, X., Wang, G., Qin, H., 2009. Surface reconstruction using bivariate simplex splines on Delaunay configurations. *Computers&Graphics*, 33, 341-350.

Kimme, C., Ballard, D. H., Sklansky, J., 1975. Finding circles by an array of accumulators. *Communications of the Association for Computing Machinery*, 18, 120-122.

Li, S., Zhu, L., Zhang, Q., Blake, A., Zhang, H. J., and Shum, H., 2002. Statistical learning of multi-view face detection. Proceedings *of the 7th European Conference on Computer Vision*, Copenhagen, Denmark.

Lübcke, H., Schmidt, S., Rung, T., Thiele, F., 2001. Comparison of LES and RANS in bluff-body flows. *Journal of Wind Engineering and Industrial Aerodynamics*, 89, 1471-1485.

Muddada S., Patnaik B.S.V., 2010. An assesment of turbulence models for the prediction of flow past a circular cylinder with momentum injection. *Journal of Wind Engineering and Industrial Aerodynamics*, 98, 575-591.

Muñoz-Salinas, R., Medina-Carnicer, R., Madrid-Cuevas, F. J., Carmona-Poyato, A., 2009. People detection and tracking using stereo vision and color. *Journal Vision Commun. Image R.*, 20, 339-350.

Shapiro, L., Stockman, G., 2001. *Computer Vision*. Prentice-Hall, Inc.

Sochman, J., Matas, J., 2004. AdaBoost with totally corrective updates for fast face detection. *Proceedings of the 6th IEEE International Conference on Automatic Face and Gesture Recognition*, 445-450.

Viola, P. and Jones, M., 2001. Rapid object detection using a boosted cascade of simple features. *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR '01)*, 1, 511-518.