MULTIDIMENSIONAL STEERABLE FILTERS AND 3D FLOW ESTIMATION

Dimitrios Alexiadis, Nikolaos Mitianoudis*

Democritus University of Thrace,
Dpt. of Electrical & Computer Engineering,
University Campus Xanthi, 67100, Greece

Tania Stathaki

Imperial College of London,
Dpt. of Electrical & Electronic Engineering,
Exhibition road, SW7 2AZ, London, U.K

ABSTRACT

In this work, the 3D flow estimation problem is formulated in the 4D spatiotemporal frequency domain, and it is shown that 3D motion manifests itself as energy concentration along hyper-planes in that domain. Based on this, the construction and use of appropriate directional multidimensional "steerable" filters, which can extract directional energy in space-time, is proposed. Steerable filters have been constructed for up to 3 dimensions. We extend the relevant mathematical definitions to multiple dimensions and formulate filter-based algorithms for 3D flow estimation. Experimental results on simulated and real data verify the efficiency of the algorithms.

Index Terms— 3D flow estimation, frequency domain, steerable filters

1. INTRODUCTION

Estimating the 3D flow from sequences of 3D data is an important task for 3D scene analysis and understanding, in many computer vision applications [1]. Although the problem of 2D flow estimation has been extensively studied [2, 3, 4, 5], only few works have dealt with 3D motion estimation directly from 4D data (3D+Time). These few approaches [6, 7, 8] are based simply on extending standard well-known differential 2D approaches, such as the Horn-Schunck [2] and Lucas-Kanade methods [3] or block-based matching, while they have been applied mainly for medical data analysis [7, 8]. Similarly, "range flow" estimation [9] extends standard 2D differential methods for computing flow from depth (partial 3D) data. On the other hand, most of the existing approaches estimate 3D flow from its 2D flow projections [1, 10], which provide a relatively fast but approximate solution. Some variational, more accurate but slower methods, that work directly in 3D (using the 2D video projections) include [11, 12].

In this paper, we address 3D flow estimation directly in 3D by formulating the problem in the 4D spatiotemporal frequency domain and constructing 4D directional steerable filters, which can be used for obtaining an efficient solution. Considering a local 4D neighborhood (3D space+time), we show that 3D motion manifests itself as energy concentration along hyper-planes in the 4D spatiotemporal frequency domain. Based on this, it is then shown that the use of appropriate 4D filters, which can extract directional energy in four dimensions, constitutes a potential solution. Our proposed solution is based on multidimensional directional "steerable" filters. Steerable and/or directional filters have been constructed for up to 3 dimensions, e.g. [13, 14, 5]. In this work, relevant mathematical definitions are extended to N dimensions.

A motivation for studying the problem in the frequency domain is that according to early studies, human motion perception mechanisms can be modeled based on frequency domain considerations [15]. The motivation behind constructing and using steerable filters is that, apart from their effectiveness, their use can provide time-efficient solutions, due to their interpolation ("steerability") property [5]. Moreover, the proposed steerable filter-based algorithms can be parallelized, so that to exploit the parallel computing capabilities of GPUs.

2. THEORETICAL DEVELOPMENTS

Consider a small neighborhood of a volumetric function, denoted as $f_0(\mathbf{x}_s)$, where $\mathbf{x}_s = [x, y, z]^{\mathrm{T}}$ is the spatial coordinates vector. Let also its evolution in time be denoted as $f(\mathbf{x}_s; t)$, such that $f(\mathbf{x}_s; 0) = f_0(\mathbf{x}_s)$. In the simplest 3D flow model, the flow in a small spatio-temporal neighborhood is approximated by a single velocity vector $\mathbf{v} = [v_x, v_y, v_z]^{\mathrm{T}}$, namely $f(\mathbf{x}_s; t) = f_0(\mathbf{x}_s - \mathbf{v}t)$.

2.1. 3D Flow in the frequency domain

Taking the 4D spatiotemporal FT of $f(\mathbf{x}_s; t)$ and using the FT shift property, it is straightforward to show that:

$$|F(\boldsymbol{\omega}_s; \omega_t)| = |\tilde{f}_0(\boldsymbol{\omega}_s)|\delta(\omega_t + \boldsymbol{\omega}_s^{\mathrm{T}} \cdot \mathbf{v}), \qquad (1)$$

where $\boldsymbol{\omega}_s$, ω_t stand for the spatial and temporal frequency, respectively and δ denotes the Dirac delta function.

Conclusion #1 - Motion Planes: The energy in the 4D (spatiotemporal) frequency domain $F(\boldsymbol{\omega}_s; \omega_t)$ is concentrated

*This work was supported by the F3SME research project (PE6 3210), implemented within the framework of the Action "Supporting Postdoctoral Researchers" of the Oper. Program "Education and Lifelong Learning", and is co-financed by the European Social Fund (ESF) and the Greek State - Special thanks to Dr. P. Daras for providing 3D data and the valuable discussions.

along a hyper-plane: $\omega_t + \boldsymbol{\omega}_s^{\mathrm{T}} \cdot \mathbf{v} = 0$. The hyper-plane's orientation gives the unknown velocity vector. Therefore, the estimation of the unknown velocity $\mathbf{v}$ can be cast as an orientation estimation problem in the 4D frequency space.

Considering the motion hyperplane $\omega_t + \boldsymbol{\omega}_s^{\mathrm{T}} \cdot \mathbf{v} = 0$ and making use of hyper-spherical coordinates [16], after a set of simple manipulations, one can conclude to:

Conclusion #2 - Motion signatures: The energy in the 4D spatiotemporal frequency domain is concentrated along the "signature":

$$M(\boldsymbol{\phi}; \mathbf{v}) := A_x(\boldsymbol{\phi})v_x + A_y(\boldsymbol{\phi})v_y + A_z(\boldsymbol{\phi})v_z + D(\boldsymbol{\phi}) = 0, \quad (2)$$

where $\boldsymbol{\phi} = [\phi_1, \phi_2, \phi_3]^{\mathrm{T}}$ with ϕ_1 ranging in $[0, 2\pi)$, ϕ_2 and ϕ_3 ranging in $[0, \pi)$ and

$$D(\boldsymbol{\phi}) = \cos\phi_3, \quad A_x(\boldsymbol{\phi}) = \sin\phi_3 \sin\phi_2 \sin\phi_1$$
$$A_y(\boldsymbol{\phi}) = \sin\phi_3 \sin\phi_2 \cos\phi_1, A_z(\boldsymbol{\phi}) = \sin\phi_3 \cos\phi_2. \quad (3)$$

2.2. Construction of N-D directional steerable filters

Denote as $\boldsymbol{\omega} = [\omega_1, \omega_2, \ldots, \omega_N]^{\mathrm{T}}$ a N-D frequency vector and as $\hat{\boldsymbol{\omega}} = \boldsymbol{\omega}/||\boldsymbol{\omega}||$ the corresponding unit-normalized vector. A N-D directional filter of order L, oriented along the unit vector $\mathbf{d} = [d_1, d_2, \ldots, d_N]^{\mathrm{T}}$ in the N-D frequency domain, is defined as

$$B_{\mathbf{d}}^L(\boldsymbol{\omega}) := (\hat{\boldsymbol{\omega}}^{\mathrm{T}} \cdot \mathbf{d})^L = ||\boldsymbol{\omega}||^{-L} \left(\sum_{n=1}^{N} \omega_n d_n \right)^L. \quad (4)$$

For simplicity, from now we drop the filter's order L from notation, wherever it is implied.

According to the multinomial expansion theorem [17], (4) can be expanded as follows:

$$B_{\mathbf{d}}(\boldsymbol{\omega}) = ||\boldsymbol{\omega}||^{-L} \sum [C(p_1, p_2, \ldots, p_N; L) \prod_{n=1}^{N} d_n^{p_n} \prod_{n=1}^{N} \omega_n^{p_n}], \quad (5)$$

where the summation runs for all combination of integers $p_1, p_2 \ldots, p_N \geq 0$ that sum up to L, i.e. $\sum_{n=1}^{N} p_n = L$. The expansion coefficients are given by:

$$C(p_1, p_2, \ldots, p_N; L) := \frac{L!}{p_1! p_2! \ldots p_N!}, \quad (6)$$

while the number of monomial terms (number of expansion coefficients) are equal to $I_0(N; L) := \begin{pmatrix} L+N-1 \\ N-1 \end{pmatrix}$.

We now define the vector $\mathbf{c}(\boldsymbol{\omega})$ of length $I_0(N; L)$:

$$\mathbf{c}(\boldsymbol{\omega}) := ||\boldsymbol{\omega}||^{-L} \left[\begin{array}{c} \\ \scriptstyle p_1, \ldots, p_N \geq 0 \\ \scriptstyle p_1 + \ldots + p_N = L \end{array} C(p_1, .., p_N; L) \prod_{n=1}^{N} \omega_n^{p_n} \right]^{\mathrm{T}}, \quad (7)$$

as well as the vector: $\mathbf{k}(\mathbf{d}) := \left[\begin{array}{c} \\ \scriptstyle p_1, \ldots, p_N \geq 0 \\ \scriptstyle p_1 + \ldots + p_N = L \end{array} \prod_{n=1}^{N} d_n^{p_n} \right]^{\mathrm{T}}$. Then, using vector notation, (5) is written as $B_{\mathbf{d}}(\boldsymbol{\omega}) = \mathbf{k}(\mathbf{d})^{\mathrm{T}} \cdot \mathbf{c}(\boldsymbol{\omega})$.

Basis filters: Consider $I \geq I_0(N; L)$ basis filters $B_{\mathbf{d}_i}(\boldsymbol{\omega})$, at the basic orientations $\mathbf{d}_i, i = 1, 2, \ldots, I$. Denote as $\mathbf{B}(\boldsymbol{\omega}) = [B_{\mathbf{d}_1}(\boldsymbol{\omega}), ..., B_{\mathbf{d}_I}(\boldsymbol{\omega})]^{\mathrm{T}}$ the basis filters' vector. Let also $\mathtt{K} = [\mathbf{k}(\mathbf{d}_1), \ldots, \mathbf{k}(\mathbf{d}_I)]^{\mathrm{T}}$ be a matrix of size $I \times I_0(N; L)$. The set of basis filters is by definition given from: $\mathbf{B}(\boldsymbol{\omega}) = \mathtt{K} \cdot \mathbf{c}(\boldsymbol{\omega})$. Solving for $\mathbf{c}(\boldsymbol{\omega})$, we get: $\mathbf{c}(\boldsymbol{\omega}) = \mathtt{K}^{-1} \cdot \mathbf{B}(\boldsymbol{\omega})$, where $\mathtt{K}^{-1}$ is the (pseudo-) inverse of $\mathtt{K}$.

Interpolation formula: Using the above definitions, one gets the interpolation formula:

$$B_{\mathbf{d}}(\boldsymbol{\omega}) = \mathbf{k}(\mathbf{d})^{\mathrm{T}} \cdot \mathtt{K}^{-1} \cdot \mathbf{B}(\boldsymbol{\omega}) = \mathbf{t}(\mathbf{d}) \cdot \mathbf{B}(\boldsymbol{\omega}) = \sum_{i=1}^{I} t_i(\mathbf{d}) B_{\mathbf{d}_i}(\boldsymbol{\omega}), \quad (8)$$

where $\mathbf{t}(\mathbf{d}) := \mathbf{k}(\mathbf{d})^{\mathrm{T}} \cdot \mathtt{K}^{-1}$ is the interpolation vector.

Conclusion #3 - Filter's steerability: A N-D directional filter of order L, oriented at an arbitrary orientation $\mathbf{d}$, defined as (4), can be interpolated ("steered") from $I \geq I_0(N; L)$ basic directional filters $B_{d_i}(\boldsymbol{\omega})$, using the interpolation formula (8). Due to the linearity of FT (or convolution in the space-time domain), the interpolation scheme of (8) holds also for the filter responses.

Table 1. Common parameters in all experiments

PARAMETER	VALUE
Num of consecutive input volume-frames	7
Directional filters' order L	2
Num of basis filters I (eq. distrib. on hypersphere)	16
Num of voxels for directional power calculation	$3 \times 3 \times 3$
4D Gaussian window - relative $\boldsymbol{\sigma}$ (voxels^3 frames)	(1,1,1,1)
Size of local neighborhood (voxels)	$3 \times 3 \times 3$
3D Gaussian window - relative $\boldsymbol{\sigma}$ (voxels^3)	$(\sqrt{2},\sqrt{2},\sqrt{2})$
Grid of ϕ_1 (start:step:end)	90 : 15 : 270
sph. angles ϕ_2	45 : 15 : 135
(degrees) ϕ_3	45 : 15 : 135

Table 2. Different parameters in experiments

	VALUE		
PARAMETER	EXP. #1	EXP. #2	EXP. #3
Size of voxels (mm^3)	15^3	15^3	40^3
Num of voxels/frame	$67 \times 67 \times 67$	$75 \times 71 \times 33$	$35 \times 48 \times 41$

3. PROPOSED METHODS

Motion signatures as motion constraints: Conclusion #2 could be reformulated as follows: Given a specific direction $\boldsymbol{\phi}^{\mathrm{m}}(\mathbf{x}) = \left[\phi_1^{\mathrm{m}}(\mathbf{x}), \phi_2^{\mathrm{m}}(\mathbf{x}), \phi_3^{\mathrm{m}}(\mathbf{x})\right]^{\mathrm{T}}$ in the hyper-spherical frequency domain, where a large portion of energy is known to be concentrated, the unknown velocity $\mathbf{v} = [v_x, v_y, v_z]^{\mathrm{T}}$ should satisfy the 3D motion constraint (linear) equation: $M(\boldsymbol{\phi}^{\mathrm{m}}; \mathbf{v}) = 0$. This single 3D motion constraint equation cannot be used alone to obtain the full 3D velocity component. The problem is ill-posed. Only the normal velocity component can be estimated (subsection 3.2). In order to estimate the full 3D velocity, multiple constraints have to be combined (subsection 3.3).

A sequence of volumetric data, i.e a 4D scalar function $f(\mathbf{x})$ constitutes the input of the proposed algorithms, where $\mathbf{x} = [\mathbf{x}_s; t] = [x, y, z, t]^{\mathrm{T}}$ is the 4D space-time vector.

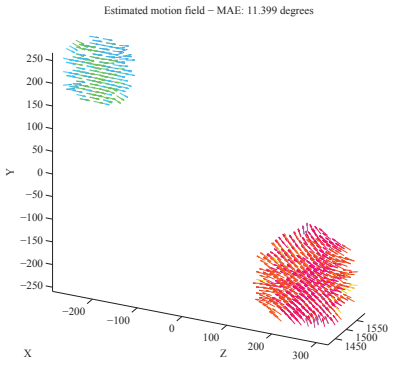

Fig. 1. Experiment #1 - Simulated "Moving spheres". The estimated 3D motion field using the proposed algorithm.

3.1. Algorithm #1: Directional spatiotemporal power

The following algorithm is used for the calculation of the "directional spatiotemporal power" (see step 6 for the definition), by efficiently exploiting the filters' "steerability" property.

1. Construct a 4D steerable filter basis in the frequency domain: $B_{\mathbf{d}_i}(\boldsymbol{\omega})$, at the basic orientations $\mathbf{d}_i, i = 1, 2, \dots I$.
2. Take the 4D spatiotamporal FT of the input $f(\mathbf{x})$, to obtain $F(\boldsymbol{\omega})$, i.e. the input in the frequency domain.
3. Multiply $F(\boldsymbol{\omega})$ with each basic filter $B_{\mathbf{d}_i}(\boldsymbol{\omega})$, to obtain the basic responses $F_i(\boldsymbol{\omega}) = B_{\mathbf{d}_i}(\boldsymbol{\omega})F(\boldsymbol{\omega}), i = 1, 2, \dots, I$.
4. Apply 4D IFT to each $F_i(\boldsymbol{\omega})$ get the basic responses $f_i(\mathbf{x}) = b_{\mathbf{d}_i}(\mathbf{x}) * f(\mathbf{x})$ in the original space-time domain.
5. Consider a dense grid of directions $\boldsymbol{\phi} = [\phi_1, \phi_2, \phi_3]^{\mathrm{T}}$. Calculate the response $f_{\boldsymbol{\phi}}(\mathbf{x})$ for each $\boldsymbol{\phi}$ in the grid, as a linear combination of the basis responses.
6. Calculate the directional spatiotemporal power $R(\boldsymbol{\phi})(\mathbf{x}) = |f_{\boldsymbol{\phi}}(\mathbf{x})|^2$. Actually, to deal with noise, a more robust functional is used, which integrates locally the power:

$$R(\boldsymbol{\phi})(\mathbf{x}) = \sum_{\mathbf{x}_n \in \mathcal{N}(\mathbf{x})} g_1(\mathbf{x} - \mathbf{x}_n)|f_{\boldsymbol{\phi}}(\mathbf{x}_n)|^2, \qquad (9)$$

where $g_1(\mathbf{x})$ is a narrow 4D separable Gaussian function with $\boldsymbol{\sigma} = [\sigma_x, \sigma_y, \sigma_z, \sigma_t]^{\mathrm{T}}$. For the experimental value of $\boldsymbol{\sigma}$ and the size of neighborhood $\mathcal{N}(\mathbf{x})$, see Table 1.

3.2. Algorithm #2: Estimation of 3D normals and normal velocity component

The 3D normals map $\mathbf{n}_s(\mathbf{x})$ at each spatiotemporal location $\mathbf{x}$ and the speeds $S(\mathbf{x})$ along the normals can be calculated by the following algorithm:

1. Using Algorithm #1, the directional power $R(\boldsymbol{\phi})(\mathbf{x})$ is extracted and the dominant 4D orientation is found: $\boldsymbol{\phi}^{\mathrm{m}}(\mathbf{x}) = \left[\phi_1^{\mathrm{m}}(\mathbf{x}), \phi_2^{\mathrm{m}}(\mathbf{x}), \phi_3^{\mathrm{m}}(\mathbf{x})\right]^{\mathrm{T}} = \arg\max_{\boldsymbol{\phi}}\{R(\boldsymbol{\phi})(\mathbf{x})\}$.
2. Using hyper-spherical to Cartesian transformation, the unit 4D direction vector $\mathbf{n}(\mathbf{x}) = \left[n_x(\mathbf{x}), n_y(\mathbf{x}), n_z(\mathbf{x}), n_t(\mathbf{x})\right]^{\mathrm{T}}$ is estimated from $\boldsymbol{\phi}^{\mathrm{m}}(\mathbf{x})$.
3. The spatial normal (plane normal) $\mathbf{n}_s(\mathbf{x})$ and the normal velocity component $S(\mathbf{x})$ are then given from (we drop $\mathbf{x}$ for simplicity): $\mathbf{n}_s = \frac{[n_x, n_y, n_z]^{\mathrm{T}}}{\sqrt{n_x^2+n_y^2+n_z^2}}$ and diagonal $S = \frac{-n_t}{\sqrt{n_x^2+n_y^2+n_z^2}}$. Actually, there are two valid solutions, with the second one given by $\mathbf{n}_s^{'} = -\mathbf{n}_s$ and $S^{'} = -S$.

3.3. Algorithm #3: Estimation of full 3D velocity

We search for the velocity $\mathbf{v}(\mathbf{x})$ that minimizes the objective function:

$$E\{\mathbf{v}(\mathbf{x})\} = \sum_{\mathbf{x}_n \in \mathcal{N}(\mathbf{x})} W(\mathbf{x} - \mathbf{x}_n) \sum_{j=1}^{J(\mathbf{x}_n)} w_j(\mathbf{x}_n)||M(\boldsymbol{\phi}^j; \mathbf{v})(\mathbf{x}_n)||_2^2 \qquad (10)$$

where $\mathcal{N}(\mathbf{x})$ is a neighborhood around $\mathbf{x}$ (of size 3x3x3 in all our experiments), $W(\mathbf{x})$ is the square of a Gaussian function (with $\boldsymbol{\sigma} = [1, 1, 1]^{\mathrm{T}}$ voxels^3 in all experiments), $w_j(\mathbf{x}_n)$ is a weight reflecting a kind of confidence about the motion constraint $M(\boldsymbol{\phi}^j; \mathbf{v})(\mathbf{x}_n) = 0$ and $J(\mathbf{x}_n)$ is a number of appropriate triplets $\boldsymbol{\phi}^j = [\phi_1^j, \phi_2^j, \phi_3^j]^{\mathrm{T}}$, for which $R(\boldsymbol{\phi}^j)(\mathbf{x})$ takes a large value. A simple method for the last one, that was used in our experiments and imposes a constant number J of motion constraints for all $\mathbf{x}$, is to consider all 2D slices $\phi_1 = \text{const}$ of $R(\phi_1, \phi_2, \phi_3)$ in the search grid and find the positions of the corresponding maxima. Finally, as for the weights $w_j(\mathbf{x}_n)$, the values of the directional power itself was used, i.e. $w_j(\mathbf{x}_n) = R(\phi^j)(\mathbf{x}_n)$.

The algorithm for the extraction of the full 3D velocity vector can be summarized as follows:

1. The directional power $R(\boldsymbol{\phi})(\mathbf{x})$ is extracted, using Algorithm #1. Then, for each voxel $\mathbf{x}$, find the triplets $\boldsymbol{\phi}^j(\mathbf{x})$, for which $R(\boldsymbol{\phi}^j)(\mathbf{x})$ takes a large value.
2. For each voxel $\mathbf{x}$, minimize the energy functional of equation (10). To do so, one has to construct the system of $N \cdot J$ linear equations:

$$W(\mathbf{x} - \mathbf{x}_n) \cdot w_j(\mathbf{x}^n) \cdot M(\phi_j^n; \mathbf{v})(\mathbf{x}_n) = 0,$$

for $j = 1, 2, \dots, J$ and $\mathbf{x}^n \in \mathcal{N}(\mathbf{x})$, i.e. all N neighbors around $\mathbf{x}$. Solving the linear system, in the least-square sense, gives the unknown velocity $\mathbf{v}$.

4. EXPERIMENTAL RESULTS

Evaluation metric: If the ground-truth (GT) flow field is known, the algorithm's performance evaluation is based on the (Mean) Angular Error (M)AE [4], extended however for the 3D flow case. An estimated 3D motion vector $\mathbf{v}$ is expressed in homogenous coordinates as $\overline{\mathbf{v}} = \frac{(\mathbf{v}^{\mathrm{T}},1)^{\mathrm{T}}}{||(\mathbf{v}^{\mathrm{T}},1)||}$, namely as a unit direction vector in 4D. The AE is then calculated from $\mathrm{AE} = \cos^{-1}(\overline{\mathbf{v}} \cdot \overline{\mathbf{v}}^{\mathrm{GT}})$.

Experimental setup: The experiments with real-world data were realized considering sequences of 3D point clouds, reconstructed using a set of calibrated Kinect sensors [18, 19]. To use point clouds in our underlying framework however, a sequence of volumetric functions has to be constructed from the input point sets. For the experiments in this paper, we use the the simplest possible approach to realize that. The

Fig. 2. Experiment #2 - Input point clouds (left), output of Algorithm #2 (middle) and Algorithm #3 (right).

3D bounding box for all input point clouds is uniformly discretized into $N_x \times N_y \times N_z$ cubic voxels. A binary (0/1) volumetric function is reconstructed by indicating a voxel as "occupied" (set to 1) if it contains at least one point. Finally, the reported execution times refer to a desktop PC, with an i7-2700K CPU, at 3.50GHz, 8GB RAM, as well as a GeForce GTX 560 GPU.

Parallel implementation: Since the proposed algorithms involve mainly voxel-wise operations, an advantage is that they can be parallelized to exploit the computing capabilities of GPUs. In this work, algorithm #1 was parallelized using CUDA [20]. All reported times refer to this implementation.

4.1. Experiments and results

In all provided results, the 3D motion vectors are plotted in the 3D space, using the Matlab function `quiver3`. In order to assist reader, each motion vector is plotted with a color, which depends on motion vectors direction in the 3D space. A HSV colormap of 16×16 distinct colors (directions) is used.

Experiment #1 - "Moving Spheres" with known GT: In this experiment, we use a simulated sequence with two moving spheres of radius 50mm and 80mm, respectively. The small sphere is moving towards right with velocity $\mathbf{v}_1 = [20, 0, 0]^{\mathrm{T}}$ mm/frame. The large one is moving towards left-up with velocity $\mathbf{v}_2 = [-20, 10, 0]^{\mathrm{T}}$. We applied Algorithm #3 for the estimation of full 3D velocities. The estimated 3D motion field is given in Fig. 1. The MAE metric is 11.39^o, which is a sensible value for dense flow estimation [4]. The execution time, given the algorithm's parameters in Tables 1 and 2, are presented in Table 3. As can be verified, despite the large amount of data in the 4D space, the execution time is quite low.

Experiment #2 "Dimitris" captured by one Kinect: A moving human ("Dimitris"), captured by one Kinect, is used in this experiment. The reconstructed point clouds for the 1st and last (7th) frame are given in Fig. 2(left). One can see that the human rotates around his vertical body axis, clockwise. Since the GT is not available, only qualitative evaluation of the proposed methodology results is possible. The output of Algorithm #2 is also given in Fig. 2(middle), i.e. the normal map (by taking the mean of normals along Z) and the estimated speeds map, which are combined into a vector field. Qualitatively, the results are sensible. Finally, the output of algorithm #3, for the estimation of full 3D velocities is given in Fig. 2(right). The execution time is reported in Table 3.

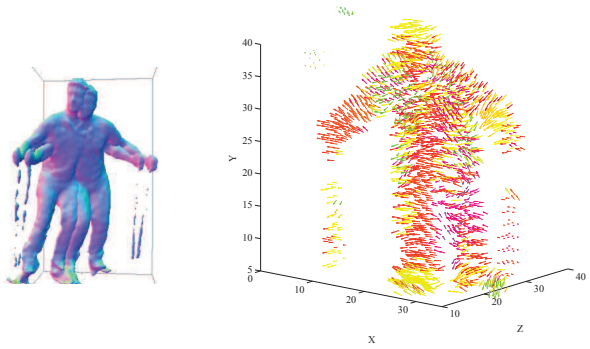

Fig. 3. Experiment #3 - "Dimitris" skiing sequence. Superimposed point clouds for the first and last frame (left) and estimated motion field (right).

Table 3. Processing time (msec) for all steerable filters-related operations

ALGORITHMIC STEP	EXP.#1	EXP.#2	EXP.#3
Calculation of responses (FT domain)	178	110	42
Inverse FT of the responses	1776	836	405
Interpolation of responses at all candidate 4D directions - Calculation of directional power	1349	1251	509
TOTAL	3303	2197	956

Experiment #3 Full 3D "Dimitris" skiing sequence: Finally, we present experimental results on a sequence of full 3D data, reconstructed using multiple Kinect sensors [18, 21]. The data were downloaded from http://vcl.iti.gr/reconstruction/ and contain a human performing skiing, as shown in Fig. 3. The output of algorithm #3 for frames 249-255 are given in Fig. 3. Studying the results in the Figure, the estimated 3D motion field seems qualitatively correct. The human moves forward-left and the left/right shoulders move upwards/downwards, respectively. This is in accordance with the estimated field. More results on this sequence can be found at http://utopia.duth.gr/%7Enmitiano/SkiingResults.pdf.

5. CONCLUSIONS

In this paper, the 3D flow estimation problem in the 4D spatiotemporal frequency domain, has been formulated. Additionally, towards providing an efficient solution, the construction of multi-dimensional steerable filters has been presented for the first time, along with algorithms for their application in 3D flow estimation. Preliminary results on simulated and real data verified the efficiency of the proposed algorithms.

6. REFERENCES

[1] S. Vedula, S. Baker, P. Rander, R. Collins, and T. Kanade, "Three-dimensional scene flow," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 27, pp. 475–480, Mar. 2005.

[2] B. Horn and B. Schunck, "Determining optical flow," *Artif. Intel.*, vol. 17, pp. 185–204, 1981.

[3] B. Lucas and T. Kanade, "An iterative image registration technique with an application to stereo vision," in *Proc. Seventh International Joint Conference on Artificial Intelligence*, 1981, pp. 674–679.

[4] J. Barron, D. Fleet, and S. Beauchemin, "Performance of optical flow techniques," *Int. J. Comput. Vision*, vol. 12(1), pp. 43–77, 1994.

[5] Dimitrios S. Alexiadis and Geroge. D. Sergiadis, "Narrow directional steerable filters in motion estimation," *Computer Vision and Image Understanding*, vol. 110(2), pp. 192–211, 2008.

[6] J. L. Barron and N. A. Thacker, *Tutorial: Computing 2D and 3D Optical Flow*, Tina Memo No. 2004-012, 2005.

[7] J. L. Barron, "Experience with 3D optical flow on gated MRI cardiac datasets," in *1st Canadian Conference on Computer and Robot Vision*, 2004, pp. 370–377.

[8] A. A. Kassim, P. Yan, W. S. Lee, and K. Sengupta, "Motion compensated lossy-to-lossless compression of 4-D medical images using integer wavelet transforms," *IEEE Trans. on Information Technology in Biomedicine*, vol. 9, pp. 132–138, Mar. 2005.

[9] Hagen Spies, Bernd Jaehne, and John L. Barron, "Range flow estimation," *Computer Vision and Image Understanding*, vol. 85, pp. 209–231, 2002.

[10] Michael B. Holte, Bhaskar Chakraborty, Thomas B. Moeslund, and Jordi Gonzalez, "A local 3D motion descriptor for multi-view human action recognition from 4d spatio-temporal interest points," *IEEE Journal of Selected Topics in Signal Processing, Special Issue on emerging techniques in 3D*, vol. 6, pp. 553 – 565, Sep. 2012.

[11] T. Basha, T. Aviv, Y. Moses, and N. Kiryati, "Multi-view scene flow estimation: A view centered variational approach," in *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2010.

[12] C. Vogel, K. Schindler, and S. Roth, "3D scene flow estimation with a rigid motion prior," in *International Conference on Computer Vision (ICCV)*, 2011.

[13] Richard P. Wildes and James R. Bergen, "Qualitative spatiotemporal analysis using an oriented energy representation," in *Computer Vision ECCV 2000, Lecture Notes in Computer Science*, 2000, vol. 1843, pp. 768–784.

[14] K.G. Derpanis and J. M. Gryn, "Three-dimensional n-th derivative of Gaussian separable steerable filters," in *IEEE International Conference on Image Processing (ICIP)*, 2005.

[15] Edward H. Adelson and James R. Bergen, "Spatiotemporal energy models for the perception of motion," *J. Opt. Soc. Am. A*, vol. 2, pp. 284–299, Feb.1985.

[16] David W. Henderson and Daina Taimina, *Experiencing geometry: on plane and sphere*, Prentice Hall, 1996.

[17] Michiel Hazewinkel, Ed., *Multinomial coefficient*, Encyclopedia of Mathematics, Springer,, 2001.

[18] D. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, full 3-D reconstruction of moving foreground objects from multiple consumer depth cameras," *IEEE Transactions on Multimedia*, vol. 15(2), pp. 339–358, Feb. 2013.

[19] D. Alexiadis, D. Zarpalas, and P. Daras, "Real-time, realistic full-body 3D reconstruction and texture mapping from multiple kinects," in *11th IEEE IVMSP Workshop*, June 2013.

[20] "Compute Unified Device Architecture," Online: https://developer.nvidia.com/category/zone/cuda-zone.

[21] Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe, "Poisson surface reconstruction," in *Symposium on Geometry Processing*, 2006, pp. 61–70.