Sparse Axes-aligned MFlux

An Efficient Alternative to Spherical Flux

A. Schuh[1], J. N. Kaftan[2], C. Tietjen[3], and T. P. O'Donnell[4]

[1] Section of Biomedical Image Analysis, Department of Radiology, University of Pennsylvania, Philadelphia, PA 19104, USA
[2] Siemens Magnet Technology, Science & Technology, Whitney OX29 4BP Oxon, UK
[3] Siemens Healthcare Sector, Computed Tomography, 91301 Forchheim, Germany
[4] Siemens Corporate Research, 755 College Rd East, Princeton, NJ 08540, USA

Abstract. The spherical flux image feature ($Flux$) serves to enhance tubular objects. Non-linear variants of $Flux$ (e.g. $MFlux$) significantly extend its power but are too computationally expensive to be applied to an entire image volume. Recently, a Fourier-based formulation of $Flux$ ($FastFlux$) was introduced which, however, must be run on an entire volume. It is unclear how to (or whether it is even possible to) incorporate non-linear characteristics. In this paper, we introduce a novel approach to flux computation which is based on two remarkable insights: First, by approximating the sphere with integral-valued vector samples, we gain a significant speed-up with only a minor degradation in response. Second, computing $MFlux$ on 3 rings aligned with the coordinate axes yields results similar to those of $Flux$. The combination of these two insights yields "Sparse Axes-aligned MFlux" ($SAMFlux$) which is faster than $FastFlux$. We demonstrate the capabilities of $SAMFlux$ in the challenging domain of liver vessel segmentation with excellent results.

Keywords: flux, medialness, vessel, tube, segmentation

1 Introduction

Flux may be defined as a scalar which describes the strength of a vector field passing through a given surface in the direction normal to that surface. In the context of image segmentation, flux is often thought of as a feature which highlights the centerlines of tubular objects. In this case, for a given voxel, the surface is a ball with the voxel at its center, and the vector field is the image gradient. The flux response has a large magnitude when the ball lies on the medial axis, its radius is the distance to the wall, and the image gradient is strong. If the radius of the tube is not known, flux must be computed on multiple scales.

Since its introduction in the vessel segmentation community by Vasilevskiy and Siddiqi [10], (multi-scale) spherical flux ($Flux$) has enjoyed popularity and spawned several variants (e.g., $CFlux$ and $MFlux$ [9], OOF [7]). However, these variants are computationally expensive and therefore have been primarily employed as needed in tracking applications, as opposed to being evaluated over

an entire image volume. Law and Chung [8] proposed a much faster method for computing $Flux$ in the Fourier domain ($FastFlux$). However, their approach *requires* flux to be computed over the entire volume and it is unclear if it can incorporate the powerful features that distinguish the variants above (e.g., the non-linearity of $MFlux$).

In this paper, we introduce a novel non-linear method for flux computation, "Sparse Axes-aligned MFlux" ($SAMFlux$) which is faster than $FastFlux$ and could be run on an entire volume. However, it is computed in the *spatial* domain, and therefore can be run in selected regions only. $SAMFlux$ is based on two unexpected observations: First, by replacing real-valued vectors sampling the surface of the sphere with rounded-off integral-valued vectors, we benefit from a significant speed-up in computation with minimal loss in response. This is due to the redundancy in the resulting samples and the elimination of the need for interpolation. Second, computing flux over 3 rings (great circles on the sphere) aligned with the coordinate axes yields similar results.

We establish the speed and effectiveness of our observations by applying it to the challenging domain of liver vessel segmentation. The vessels of the liver perfuse through the entire organ and are manifold in number; a typical venous CT Angiography (CTA) of a liver highlights well over 100 vessels of varying sizes. We applied our method to the entire liver (and only the liver) and compared it to $Flux$ (and $FastFlux$) and its variants $CFlux$ and $MFlux$.

2 Related Work

The vessel-enhancing features in the literature fall broadly into three categories: hessian-based, ray casting-based, and gradient-based (see [9] for an excellent overview of this topic). The hessian-based methods [2] have the disadvantage that they are sensitive to noise. This is due to the potential inclusion of neighboring structures from the computation of second derivatives. In addition, eigen decomposition for those methods requiring it is computationally expensive. The ray casting-based techniques [3] have high accuracy, but generally also require long computation times. The last category contains $Flux$ and its variants.

$Flux$ and its variant $CFlux$ may be described by the same equation (1). While $Flux$ is evaluated over a sphere (cf. Fig 1b), $CFlux$ is evaluated over a ring (cf. Fig 1c). $CFlux$ was designed for tracking: the intention is to evaluate it in directions, $\boldsymbol{d}$, to determine the local path of the tubular object. At position, $\boldsymbol{p}$, $Flux$ or $CFlux$ may be written as (ignoring $\boldsymbol{d}$ for the $Flux$ formulation)

$$(C)Flux\left(\boldsymbol{p};\boldsymbol{d},r\right)=\frac{1}{N}\sum_{i=1}^{N}\left\langle \nabla I_{\sigma}\left(\boldsymbol{p}+\boldsymbol{x}_{i}^{\boldsymbol{d},r}\right),-\boldsymbol{n}_{i}^{\boldsymbol{d},r}\right\rangle \tag{1}$$

where ∇I_{σ} is the image gradient at a scale σ and N the number of "boundary samples", $(\boldsymbol{x},\boldsymbol{n})_{i}^{\boldsymbol{d},r}$, where $\boldsymbol{x}$ is the boundary offset relative to the center of the sphere with radius r and $\boldsymbol{n}$ is the outward surface normal (cf. Fig 1a).

$MFlux$ (the "M" stands for minimal) was designed to avoid high flux responses when only one side of the ring was in contact with a gradient (this could

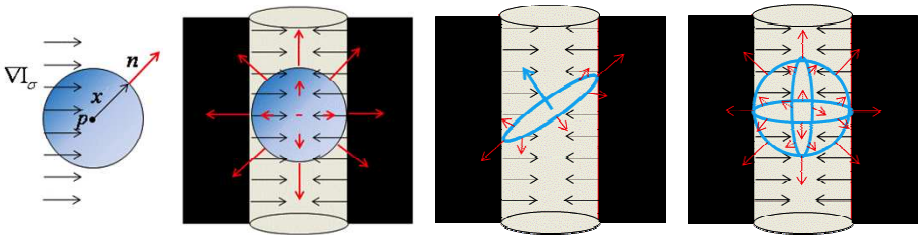


Fig. 1: (a) Components of Eqs 1 & 2. (b) *Flux*. (c) *CFlux*. (d) *SAMFlux*.

happen when that gradient was very strong). Therefore, the $MFlux$ score is the sum of the minima of the pairs $\langle \nabla I_\sigma (\boldsymbol{p}+\boldsymbol{x}), -\boldsymbol{n} \rangle$ on opposite points. In this sense, it may be considered non-linear. $MFlux$ may hence be written as

$$MFlux(\boldsymbol{p};\boldsymbol{d},r) = \frac{2}{N}\sum_{i=1}^{\frac{N}{2}} \min\left(\left\langle \nabla I_\sigma\left(\boldsymbol{p}+\boldsymbol{x}_i^{\boldsymbol{d},r}\right), -\boldsymbol{n}_i^{\boldsymbol{d},r}\right\rangle, \left\langle \nabla I_\sigma\left(\boldsymbol{p}+\boldsymbol{x}_{i+\frac{N}{2}}^{\boldsymbol{d},r}\right), -\boldsymbol{n}_{i+\frac{N}{2}}^{\boldsymbol{d},r}\right\rangle \right) . \tag{2}$$

3 Method

We introduce two approaches for increasing the efficiency of flux. The first applies to any non-Fourier-based flux feature, for example, *Flux*, *CFlux*, or *MFlux*. In this first approach, as opposed to conventional flux, we replace real-valued offsets $\boldsymbol{x}$ by integer-valued offsets $\tilde{\boldsymbol{x}}$ corresponding to the nearest voxel positions. Normals, $\boldsymbol{n}$, with values equal to specific $\boldsymbol{x}$'s are recomputed as $\tilde{\boldsymbol{n}} = \tilde{\boldsymbol{x}}/\|\tilde{\boldsymbol{x}}\|$. This discretization of offsets and normals results in different values of $(\boldsymbol{x},\boldsymbol{n})$ rounding off to the same values $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{n}})$. The resulting redundancies may come from the different positions on the same sphere (ring) or even different spheres (rings). Therefore, we identify the set of "unique boundary samples", $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{n}})_k$, for a particular $\boldsymbol{p}$ and compute the dot products at these samples only once. Most often, this reduces the number of dot product computations significantly. The computation of flux is now divided into two steps as shown in Alg 1: Step 1, the computation of dot product values; Step 2, the evaluation of the sum in either (1) or (2) at different scales, r (and in multiple directions, d, for CFlux or MFlux). For each discrete sample $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{n}})_i^{\boldsymbol{d},r}$, a reference to the memory for the dot product value, dp_k, to be evaluated at the unique sample $(\tilde{\boldsymbol{x}},\tilde{\boldsymbol{n}})_k$ with equal offset $\tilde{\boldsymbol{x}}$ is reserved. Another advantage over the conventional implementation is that Step 1 can easily be accelerated using Single Instruction, Multiple Data (SIMD) operations such as the Streaming SIMD Extensions (SSE) [4]. This allows the evaluation of four dot products at a time.

Alg. 1 Efficient Implementation for Flux Computation.

// Step 1: Evaluate dot products (four at a time using SSE).
for *each unique boundary sample* $(\boldsymbol{x}, \boldsymbol{n})_k$ **do**
    $dp_k = \langle \nabla I_\sigma(\boldsymbol{p} + \boldsymbol{x}_k), \boldsymbol{n}_k \rangle$
    (dp_k is stored in pre-allocated list)
end
// Step 2: Evaluate flux.
$flux\,(\boldsymbol{p}) = -\infty$
for *each direction* $\boldsymbol{d}$ *and radius* r **do**
    $tmp = (C|M)Flux\,(\boldsymbol{p}; \boldsymbol{d}, r)$
    (where the in Step 1 (pre-)computed dot product values dp_k are used)
    if $tmp > flux(\boldsymbol{p})$ **then**
        $flux(\boldsymbol{p}) = tmp$
    end
end

The second approach to increase the efficiency of flux applies to $Flux$ (as well as $CFlux$ and $MFlux$ when the latter two are evaluated in multiple directions to cover an entire sphere). By evaluating flux in just three directions aligned with the coordinate axes, we are able to achieve results similar to a more densely sampled evaluation. Therefore, we define a new variant of $Flux$ named $SAMFlux$. In order for it to be robust against noise and the presence of other, non-tubular structures, we utilize the non-linear $MFlux$ and thus write $SAMFlux$ as

$$SAMFlux\,(\boldsymbol{p}; r) = \max_{\boldsymbol{d} \in \{\boldsymbol{e}_x, \boldsymbol{e}_y, \boldsymbol{e}_z\}} MFlux\,(\boldsymbol{p}; \boldsymbol{d}, r) \tag{3}$$

where $\boldsymbol{e}_x$, $\boldsymbol{e}_y$, and $\boldsymbol{e}_z$ are the orthogonal basis vectors of the Cartesian coordinate system of the image volume. In other words, $SAMFlux$ is the $MFlux$ feature (with integral vector efficiencies described above) evaluated in the axial, coronal, and sagittal viewing directions, where the three oriented responses are combined using the maximum operator. Multi-scale $SAMFlux$ calculated over a set of radii $\mathcal{R}$ is computed as

$$SAMFlux\,(\boldsymbol{p}) = \max_{r \in \mathcal{R}} SAMFlux\,(\boldsymbol{p}; r) \qquad . \tag{4}$$

4 Results

We evaluated the flux-based features at multiple scales (radii 2, 3, 4, and 5 mm) on 43 CTA liver datasets of varying quality, extent of disease, and protocol. All scans were routinely acquired on Siemens equipment at different clinical sites. Slice thickness ranged from 0.8-1.5 mm. We computed all gradient images at a scale σ of 2 mm after the liver was segmented and areas of liver lesions and intense Lipiodol uptake were removed automatically [5]. Fig 2 compares the feature responses on an exemplary coronal slice, where 39 orientations, $\boldsymbol{d}$, were

considered in case of $CFlux$ and $MFlux$. It is clearly observable that $SAMFlux$ is similar to $Flux$, but is more concentrated near the medial axis.

(a) (b) (c) (d) (e)

Fig. 2: (a) Image intensities (with lesion near bottom removed). (b) Maximum $Flux$. (c) Maximum $CFlux$. (d) Maximum $MFlux$. (e) Maximum $SAMFlux$.

To judge relative speed, we compared running times measured for 10 volumes on a machine equipped with an Intel Xeon X5570 CPU and 18 GB of RAM. Running times of the conventional implementation (which tri-linearly interpolates the gradient field) and our proposed implementation (which uses integer-valued boundary offsets and no interpolation) are summarized for the different flux variants in Table 1b. Table 1a demonstrates the reduction in the number of dot product computations when employing our proposed efficiencies. Table 2 compares the running times of our proposed implementation of $Flux$ to both the conventional implementation and $FastFlux$. It is important to note that both our proposed and the conventional implementation are run on the liver only, processing whole slices by separate threads, whereas the Fourier-based $FastFlux$ needs to be computed over the entire volume (i.e., the bounding box of the liver).

conventional	proposed
3058	750
839	623
445	415
248	232

(a)

feature	PI	CI	$\frac{PI}{CI} \cdot 100\%$
Flux	0.4438	2.495	17.79%
CFlux	1.975	20.52	9.625%
MFlux	2.160	23.01	9.387%
SAMFlux	0.2582	1.623	15.91%

(b)

Table 1: (a) The number of dot product computations necessary when using our proposed approach versus the conventional approach. (Total numbers of boundary samples on spheres with radii 1, 2, 3, 4, and 5 mm sampled with different accuracies; voxel size 1x1x2 mm^3.) (b) Running times (in seconds) per 512x512 voxels based on proposed (PI) and conventional (CI) implementation.

Having established the speed of $SAMFlux$, we turn our attention to evaluating the relative quality of its response. We performed an assessment of feature

volume size	voxel size [mm^3]	total running times				
		PI	FI	CI	$\frac{PI}{FI} \cdot 100\%$	$\frac{PI}{CI} \cdot 100\%$
224 x 289 x 189	0.71 x 0.71 x 0.80	7.137	20.97	35.20	34.03%	20.28%
240 x 286 x 195	0.69 x 0.69 x 0.80	7.565	18.49	36.88	40.91%	20.51%
316 x 289 x 162	0.64 x 0.64 x 0.99	8.731	26.64	44.08	32.77%	19.81%
339 x 255 x 189	0.66 x 0.66 x 0.90	8.811	27.09	38.96	32.52%	22.62%
325 x 258 x 197	0.65 x 0.65 x 0.80	9.559	28.56	42.75	33.47%	22.36%
290 x 245 x 229	0.71 x 0.71 x 0.92	9.378	38.27	45.81	24.50%	20.47%
274 x 252 x 222	0.66 x 0.66 x 0.80	9.201	29.44	47.11	31.25%	19.53%
297 x 267 x 241	0.66 x 0.66 x 0.80	10.49	35.16	54.83	29.84%	19.13%
354 x 273 x 190	0.53 x 0.53 x 0.80	11.52	37.29	58.12	30.89%	19.82%
348 x 247 x 204	0.59 x 0.59 x 0.84	10.28	35.03	47.78	29.35%	21.52%
				average	31.95%	20.61%

Table 2: Running times (in seconds) for $Flux$ computation based on proposed (PI), conventional (CI), and $FastFlux$ implementation (FI).

responses on a centerline level and used expert annotated centerlines as ground-truth from 43 datasets. In the following, we denote the annotated centerline of a dataset by $\mathcal{A}$, the set of voxels which represents it in the image domain.

For each dataset and flux variant, we thresholded the feature response volume to remove the worst $t \cdot 100\%$ of positive responses, and varied t to obtain different binarizations. For each binarization, we computed a skeleton $\mathcal{S}_t$ of voxels using a thinning method [6] and compared it to the reference skeleton $\mathcal{A}$. We therefore determined for each voxel $\boldsymbol{p} \in \mathcal{S}_t$ the distance $d_{\mathcal{A}}(\boldsymbol{p}) = \inf_{\boldsymbol{q} \in \mathcal{A}} d(\boldsymbol{p}, \boldsymbol{q})$, where $d(\boldsymbol{p}, \boldsymbol{q})$ is the Euclidean distance, and similarly for each voxel $\boldsymbol{q} \in \mathcal{A}$ the distance $d_{\mathcal{S}_t}(\boldsymbol{q}) = \inf_{\boldsymbol{p} \in \mathcal{S}_t} d(\boldsymbol{p}, \boldsymbol{q})$. Given these minimal distances and a maximally tolerable distance d_{max} of 2 mm, we determined the numbers

$$TP_t = |\{\boldsymbol{p} \in \mathcal{S}_t \mid d_{\mathcal{A}}(\boldsymbol{p}) \leq d_{max}\}| \quad \text{(true positives)}, \tag{5}$$

$$FP_t = |\{\boldsymbol{p} \in \mathcal{S}_t \mid d_{\mathcal{A}}(\boldsymbol{p}) > d_{max}\}| \quad \text{(false positives)}, \tag{6}$$

$$FN_t = |\{\boldsymbol{q} \in \mathcal{A} \mid d_{\mathcal{S}_t}(\boldsymbol{q}) > d_{max}\}| \quad \text{(false negatives)}. \tag{7}$$

Note that a number of true negatives cannot be defined properly as one-voxel wide skeletons are compared. Therefore, we used the traditional effectiveness measures precision (positive predictive value) and recall (true positive rate) to compute the F_1-score of $\mathcal{S}_t$ (extracted by the given feature from the current dataset) [1]. We determined an average F_1-score for each feature and each threshold t by averaging the numbers given by (5), (6), and (7) over all datasets.

The obtained average F_1-score of $SAMFlux$ is plotted over t in Fig 3. The average F_1-scores achieved by the conventional implementations of $Flux$, $CFlux$, and $MFlux$ are shown as well for comparison, where 39 orientations, $\boldsymbol{d}$, were considered in case of $CFlux$ and $MFlux$. From Fig 3 it is noticeable that $SAMFlux$ is similar to $Flux$. $SAMFlux$ obtains its maximum of 0.73 at $t = 92\%$ and $Flux$ its maximum of 0.72 at $t = 91\%$. It can be observed that $SAMFlux$ achieves better results for $t \leq 92\%$ than $CFlux$ and $MFlux$. This shows that $SAMFlux$

is less dependent on t, which in turn is an indicator of its strength as a vessel discriminator. These observations agree with Fig 2, which suggests that $CFlux$ as well as $MFlux$ suffers more from noise. Based on our experiments, $SAMFlux$ is applied in the clinic with a value of $t = 92\%$.

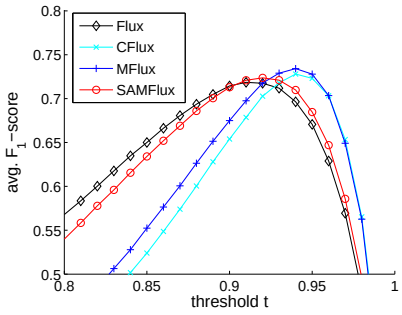


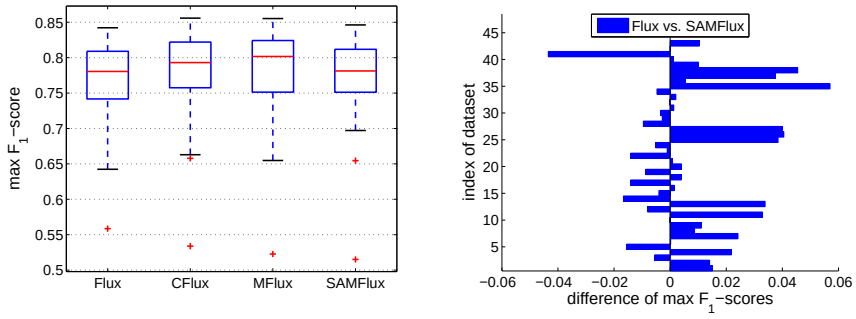
Fig. 3: Average F_1-scores (describing the average quality of the vessel discrimination) over a range of threshold values. Note that $SAMFlux$ behaves similar to $Flux$ and only slightly worse than $MFlux$.

To also study the best possible discriminative behavior of each feature, we compared the individual maximum F_1-scores achieved by them. This corresponds to having an optimal adaptive threshold selection method at hand, which would determine for each dataset the optimal threshold t. Box plots of these maxima are shown in Fig 4a. Again it can be observed that $SAMFlux$ obtains an average similar to $Flux$, but it is only slightly worse than $MFlux$ and more robust than $Flux$ and the other variants. These observations are further emphasized by the direct comparison of $SAMFlux$ and $Flux$ in Fig 4b, where a positive difference indicates $SAMFlux$'s advantage for the particular dataset.

5 Conclusions

In this paper, we presented a method for dramatically increasing the efficiency of non-Fourier-based implementations of flux as well as a novel flux formulation based on this termed $SAMFlux$. Our proposed implementation of $Flux$ [10] was shown to run at 21% of the time needed for a conventional implementation and 31% of the time needed for the Fourier-based $FastFlux$. Computation of $SAMFlux$ was found to require 0.26 sec on average per 512x512 slice while our efficient proposed version of $Flux$ required 0.44 sec. Yet, $SAMFlux$ was found to deliver comparable results and was slightly more robust to noise.

Our contributions allow for the inclusion of powerful non-linear characteristics into flux and facilitate its application on selected regions of interest in an image volume due to computation in the spatial (versus Fourier) domain. Given



(a) (b)

Fig. 4: (a) Box plots of maximum F_1-scores (describing the best possible quality of the vessel discrimination). (b) $Flux$ versus $SAMFlux$; a positive difference indicates an advantage of $SAMFlux$, a negative difference an advantage of $Flux$.

the speed and quality exhibited, flux may now be applied over an entire image volume thus opening up new possibilities for the segmentation of tubular objects in a clinical setting.

References

1. Fawcett, T.: An introduction to ROC analysis. Pattern Recogn Lett 27(8), 861–874
2. Frangi, A.F., Frangi, R.F., Niessen, W.J., Vincken, K.L., Viergever, M.A.: Multiscale vessel enhancement filtering. In: MICCAI '98. LNCS, vol. 1496, pp. 130–137. Springer-Verlag
3. Gülsün, M.A., Tek, H.: Robust vessel tree modeling. In: MICCAI '08. LNCS, vol. 5241, pp. 602–611. Springer-Verlag
4. Intel Corporation: Programming with Streaming SIMD Extensions (SSE), Intel 64 and IA-32 Architectures Software Developer's Manual, vol. 1: *Basic Architectures*, chap. 10, pp. 299–320
5. Kaftan, J.N., Tek, H., Aach, T.: A two-stage approach for fully automatic segmentation of venous vascular structures in liver CT images. In: Pluim, J.P.W., Dawant, B.M. (eds.) Medical Imaging 2009: Image Processing. vol. 7259, pp. 725911–1–12. SPIE Press
6. Kiraly, A.P., Helferty, J.P., Hoffman, E.A., McLennan, G., Higgins, W.E.: Three-dimensional path planning for virtual bronchoscopy. IEEE TMI 23(9), 1365–1379
7. Law, M.W.K., Chung, A.C.: Three dimensional curvilinear structure detection using optimally oriented flux. In: ECCV '08. pp. 368–382. Springer-Verlag
8. Law, M.W.K., Chung, A.C.S.: Efficient implementation for spherical flux computation and its application to vascular segmentation. IEEE TIP 18(3), 596–612
9. Lesage, D., Angelini, E.D., Bloch, I., Funka-Lea, G.: Design and study of flux-based features for 3D vascular tracking. In: ISBI '09. pp. 286–289
10. Vasilevskiy, A., Siddiqi, K.: Flux maximizing geometric flows. IEEE PAMI 24(12), 1565–1578