Comment faire, avec un maximum de difficultés, un filtre numérique qui non seulement ne fait rien mais le fait aussi plus lentement que si on ne l'utilisait pas !

Didier Demigny
IUT de Lannion, Département réseaux & télécommunications
Email : didier.demigny@univ-rennes1.fr
Av. Edouard Branly BP 30219, 22302 Lannion Cedex

Résumé—

A l'heure où la princesse de Clèves
s'invite en place de grève,
où des universitaires se disent,
que le savoir n'est pas une marchandise,
dans un esprit, j'espère pas trop débile
élaborons un filtre fut-il inutile
et, usant de son pouvoir bien compris
calculons nos moyennes au meilleur prix
puis, du temps perdu à la recherche
(vous verrez quelle ne coûte pas lerche)
gaussons nous d'une réponse impulsionnelle
qui nous vernira d'un lissage exponentiel.

même si tout ça ne remplit pas le bas de laine de Proust !

Décodage : on construit un filtre numérique qui ne fait rien en associant en cascade dérivateur et intégrateur. A cause du pôle placé sur le cercle unité, initialisation bien pensée, absence d'erreurs de calcul, choix judicieux de représentation des nombres sont nécessaires à une réalisation réussie. C'est facile à comprendre et à tester puisque les calculs sont réduits au minimum : une addition et une soustraction. On montre alors que la même structure de filtre légèrement modifiée permet de calculer des moyennes glissantes à coût de calcul indépendant du nombre d'éléments pris en compte dans le moyenne. Finalement, (un petit pas dans le domaine de la recherche), on généralise aux réponses impulsionnelles polynômiales avec l'exemple d'une approximation récursive efficace en rapidité du filtre gaussien.

I. Introduction

A. Quoi, à qui et comment ?

Il s'agit de traitement numérique du signal mais qui peut dériver vers des exercices de représentation des nombres, de programmation ou d'implantation numérique FPGA. Les premières victimes ont été les étudiants de $2^{\text{ème}}$ année du département *Réseaux & Télécommunications* de l'IUT de Lannion qui se destinent à des poursuites d'études. Le contenu est cependant bien adapté à des étudiants de génie électrique et peut être même d'informatique.

La section II présente une réalisation originale du filtre d'entrée x et de sortie y réalisant $y = x$. La section III utilise le principe de la section II pour faire enfin œuvre d'utilité en proposant une réalisation d'un calcul de moyenne glissante. La section IV présente une réalisation récursive d'une approximation polynômiale de filtre gaussien, toujours selon le principe de la section II. Cette réalisation est issue de travaux de recherche personnels sur les détecteurs de contours optimaux réalisés jadis [1] ayant pour origine les travaux de Canny [2] et Deriche [3]. Ces travaux sont résumés section V.

J'utilise depuis longtemps l'exemple de la moyenne (section III) comme exemple simple de filtre (version non récursive et récursive). J'ai expérimenté en cours sur un coin de tableau, initialement pour détendre mes étudiants et aussi pour les provoquer... le filtre qui ne fait rien (section II). Généralement frustrés de ne pas savoir à quoi servent les outils qu'on leur enseigne (transformées en Z etc.) ils étaient là plutôt dans la peau du quidam interloqué par le fait qu'un chercheur passe son temps sur un objet totalement inutile ! Ils furent alors très réceptifs à l'ébauche d'une explication du filtre polynomial (section IV). J'ai récidivé cette année en m'appuyant cette fois-ci sur un ensemble de dia. L'attention des étudiants et l'échange de questions valident l'approche choisie.
Si ce thème a fait l'objet d'une présentation orale au Colloque National de la Recherche des IUT en 2009, il n'a jamais été publié sous la forme d'article mis à part le contenu de la section IV.

Les sujets de TD et et de mini-projet relatifs à la section IV sont téléchargeables à l'url suivante : [4]. On trouvera en annexe le programme matlab conçu par deux étudiants : Thierry Berthou et Sébastien Tanguy (2ème année R&T) pour ce mini-projet.

B. De la recherche à la pédagogie

Je suis convaincu du rôle pédagogique essentiel des chercheurs (quel que soit leur statut) dès le premier cycle universitaire (et même dans les cycles précédents). Ce qui m'a intéressé ici c'est de montrer comment il est possible de s'inspirer d'un travail de recherche pour réaliser une séquence pédagogique. Il a fallu pour ça s'affranchir de la démarche scientifique initiale, de sa chronologie ; s'affranchir aussi des objectifs initiaux de cette recherche afin de construire du pédagogique du simple au compliqué en générant au passage

Article publié par EDP Sciences et disponible sur le site http://www.j3ea.org ou http://dx.doi.org/10.1051/j3ea/2010015

un maximum de questions réponses avec les étudiants. Ceci étant fait, il devient alors possible, en fin de cours, de leur expliquer qualitativement quelle a été la démarche du chercheur, quelles erreurs il a commises, quels problèmes il a rencontrés.

II. Le filtre qui fait néant

Ce qui va nous occuper dans cette section, c'est de réaliser un filtre tc d'entrée $x(t)$ et de sortie $y(t)$ qui calcule :

$$y(t) = x(t)$$

Comme cela est trop compliqué, on décide de décomposer le problème en deux étapes plus simples et donc d'utiliser deux filtres élémentaires. Le premier filtre d d'entrée $x(t)$ et de sortie $u(t)$ produit en sortie la dérivée du signal d'entrée.

$$u(t) = x'(t) = \frac{d\,x(t)}{dt}$$

La fonction de transfert du filtre d est :

$$D(j\omega) = j\omega$$

Le second filtre i d'entrée $u(t)$ et de sortie $y(t)$ en cascade avec le premier produit en sortie l'intégrale du signal d'entrée.

$$y(t) = \int_{t_0}^{t} u(\tau)d\tau$$

La fonction de transfert du filtre i est :

$$I(j\omega) = \frac{1}{j\omega}$$

On a donc bien :

$$TC(j\omega) = j\omega \cdot \frac{1}{j\omega} = 1$$

Mais un esprit rigoureux aura noté qu'on obtient :

$$y(t) = x(t) - x(t_0)$$

et donc le traitement souhaité à une constante près.

A. Souvenez vous de l'électronique de première année

En fait, je commence plutôt par présenter le dessin de la figure 1 en demandant quelles fonctions sont réalisées ? Sur le papier, ça fonctionne !

$$TCa(j\omega) = -jRC\omega \cdot \frac{-1}{jRC\omega} = 1$$

et le questionnement est riche. En principe, on réalise bien $y = x$ quelles que soient les conditions initiales de charge des condensateurs, mais si on inverse la position de l'intégrateur et du dérivateur, est-ce encore vrai ? De toute façon, pratiquement, on sait que les courants d'offset vont amener l'intégrateur vers la saturation, et que le montage ne fonctionnera pas. Mais alors, qu'en est-il en numérique ?

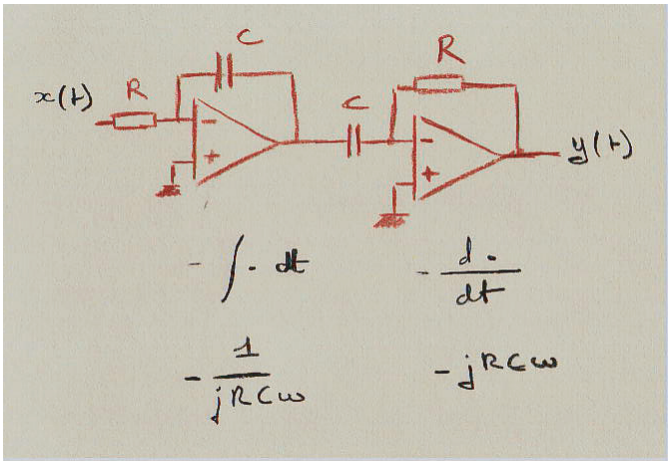

FIG. 1. Version analogique du filtre qui ne fait rien.

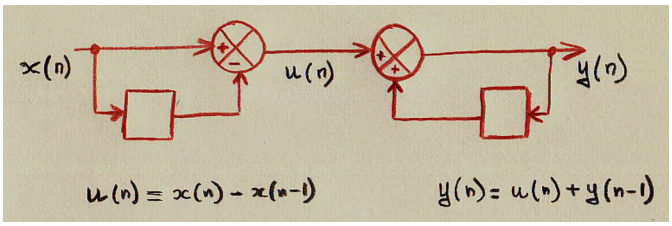

FIG. 2. Version numérique du filtre qui ne fait rien, dérivateur en tête.

B. Version numérique n°1

Une première version numérique est donnée figure 2. Le dérivateur et l'intégrateur numériques ont été vus précédemment en cours et en TD. Les carrés (registres) introduisent des retards d'un échantillon. les équations écrites figure 2 sont alors évidentes. On peut demander aux étudiants de vérifier que l'on obtient bien $y(n) = x(n)$ à une constante près et leur faire effectuer un test en étudiant les valeurs successives de $u(n)$ et $y(n)$ pour un signal d'entrée constant.

On a aussi facilement d'après le schéma [1] :

$$U(r) = X(r) - X(r-1) = X(r) - rX(r)$$

$$Y(r) = U(r) + Y(r-1) = U(r) + rY(r)$$

Soit :

$$TCN(j\omega) = (1 - e^{-j\omega}) \cdot \frac{1}{(1 - e^{-j\omega})}$$

C. Version numérique n°2, magique !

On sait que l'ordre des composants ne change pas la fonctionnalité du système linéaire (en théorie) ... On place maintenant l'intégrateur (sommateur) en tête (figure 3). Imaginons qu'on injecte toujours en entrée la suite constante $x(n) = 1, \forall n.\ u(n)$ s'incrémente à chaque itération et la sortie vaut (en théorie) toujours $y(n) = 1$. Mais pratiquement, qu'en

[1] je préfère poser $r = z^{-1}$, plus intuitif.

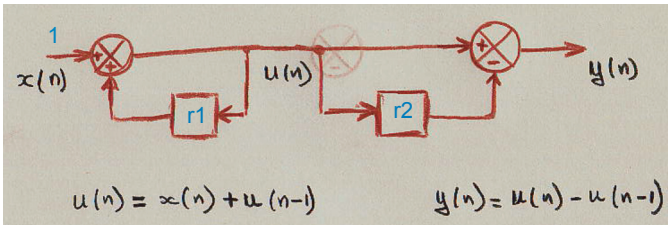

FIG. 3. Version numérique du filtre qui ne fait rien, intégrateur en tête.

est-il avec un ordinateur ? Je vous propose alors d'amener les étudiants à découvrir les problèmes de saturation dus à la représentation des nombres en machines ...
Avec une représentation non signée sur 4 bits, on obtient la succession suivante :

n	x	r1	u	r2	y
14	1	14	15	14	1
15	1	15	0	15	erreur

Par contre le complément à 2 semble faire des miracles ...
[2] Supposons que x est en complément à deux sur 4 bits. Alors $x \in [-8, +7]$. Puisque $y = x$, $y \in [-8, +7]$ est aussi codé en complément à 2 sur 4 bits. Donc, tous les calculs intermédiaires sont restreints à 4 bits en complément à 2. Ce qui donne :

n	x	r1	u	r2	y
6	1	6	7	6	1
7	1	7	-8	7	**1**
8	1	-8	-7	-8	**1**

En effet, on vérifiera par un calcul binaire que pour $n = 7$, $-8 - 7 = 1$ sur 4 bits en complément à 2 et que pour $n = 8$, $-7 + 8 = 1$.
De quoi faire un petit exercice de programmation ... Et tout ça pour ne rien faire, pas si sûr ! !

III. HISTOIRE DU MOYENNAGE !

A. Est-ce que la terre se réchauffe en Bretagne ?

Un des rares jours pluvieux en Côtes d'Armor, j'ai récupérer sur le web un relevé des températures moyennes mensuelles (noté x dans les équations ci-dessous) dans la région de St Brieuc (figure 4a). On remarque aisément les oscillations de température dues au rythme des saisons. Mais supposons que l'on cherche à savoir si la terre s'est réchauffée sur cette période de 44 mois. Il faudrait pour cela s'affranchir des variations saisonnières. Une solution simple consiste à calculer la température moyenne sur un an ; calcul qui à l'aide des échantillons que nous possédons peut être fait chaque mois (résultat y). On calcula alors naturellement (à une division par 12 près) :

$$y(n) = x(n)+x(n-1)+x(n-2)+\ldots+x(n-10)+x(n-11)$$

Le résultat est visible figure 4b. Bien entendu, le problème du transitoire et donc de l'initialisation est encore ici riche de discussion car plus visible. Quelle option choisir ?

[2] Le sage travaille en nombre signé complément à 2 ...
les résultats intermédiaires sont dimensionnés à la taille du résultat final

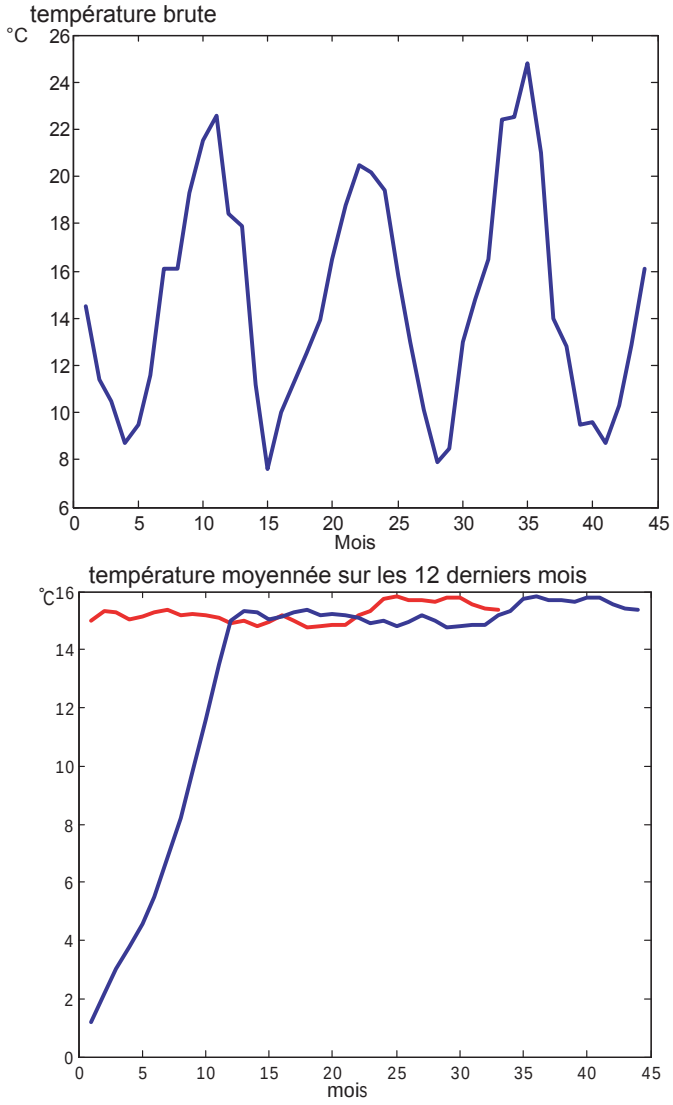

FIG. 4. (a) Températures moyennes mensuelles en Bretagne sur 44 mois. (b) Température moyenne des 12 derniers mois, calculée mensuellement

- Initialiser à 0, les échantillons d'indice négatif, c'est obtenir un transitoire de valeurs aussi inutilisables qu'inutiles (figure 4b courbe bleue).
- Considérer que les 11 premiers calculs sont forcément erronés et ne pas les prendre en compte (figure 4b courbe rouge).

On voit que l'exemple permet de donner du sens au calcul, qu'il n'y a pas de règle générale pour les initialisations et qu'il faut s'interroger sur la signification des données, des choix et des résultats.

B. Réduisons les efforts nécessaires au calcul de cette moyenne

On remarque que le calcul d'une moyenne sur les douze dernières valeurs requiert 11 additions et une division par 12 (pour considérer les 1000 dernières valeurs : 999 additions et une division). Le temps de calcul est donc proportionnel au

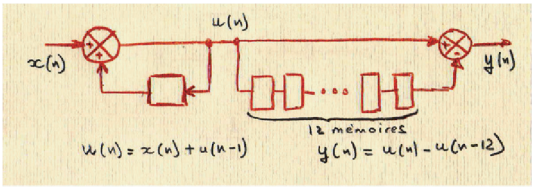

FIG. 5. Implantation récursive d'un calcul de moyenne des douze derniers échantillons d'entrée.

nombre de valeurs à prendre en compte. Il est pourtant possible de réaliser ce calcul à l'aide d'une addition et d'une soustraction uniquement. Il suffit d'écrire successivement l'expression de $y(n)$ et celle de $y(n-1)$, de les comparer pour s'apercevoir que :

$$y(n) = y(n-1) + x(n) - x(n-12)$$

On peut aussi par la transformée en r et par le calcul de la somme noter que :

$$Y(r) = X(r) \cdot (1 + r + r^2 + ... + r^{11}) = X(r) \cdot \frac{1 - r^{12}}{1 - r}$$

On remarque alors que le schéma fonctionnel de calcul de la figure 5 ressemble fortement au schéma du filtre inutile (figure 3), seul le nombre de retards avant soustraction change. On peut alors poser un certain nombre de questions aux étudiants :

- Si la température varie de -50°C à +50 °C, comment coder x ?
- Comment alors coder y ?
- et les calculs intermédiaires ?
- Comment faire la division par 12 ?
- Comment coder le résultat après division ?

IV. UNE APPLICATION SÉRIEUSE

A. Filtre gaussien

On trouvera une description complète de cette étude du point de vue recherche dans [5]. Le filtre gaussien possède des propriétés de lissage reconnues. sa réponse impulsionnelle $g^\sigma(n)$ s'écrit : $g^\sigma(n) = C_g.e^{-\frac{n^2}{2\sigma^2}}$ Il filtre d'autant plus que la largeur du filtre σ augmente. Les approximations à réponse impulsionnelle finie utilisent (pour donner un résultat satisfaisant) un nombre de coefficients de la réponse impulsionnelle d'au moins 6σ, ce qui, comme pour la moyenne, induit des temps de calcul proportionnels à σ. On sait aussi qu'il n'existe pas d'équation récursive exacte du filtre gaussien.

Ce que nous proposons, c'est une approximation de la réponse impulsionnelle du filtre gaussien par un polynôme de degré 4 (filtre POAG) avec un nombre de coefficients égal à $2W + 1$. A chaque valeur de W correspond une gaussienne approximée avec $\sigma \simeq 0,3217 \times W + 0,81$. A titre d'exemple, la figure 6 montre l'approximation pour $W = 20$ [3].

[3] l'erreur quadratique moyenne reste inférieure à 6% pour $W \in [1, 20]$

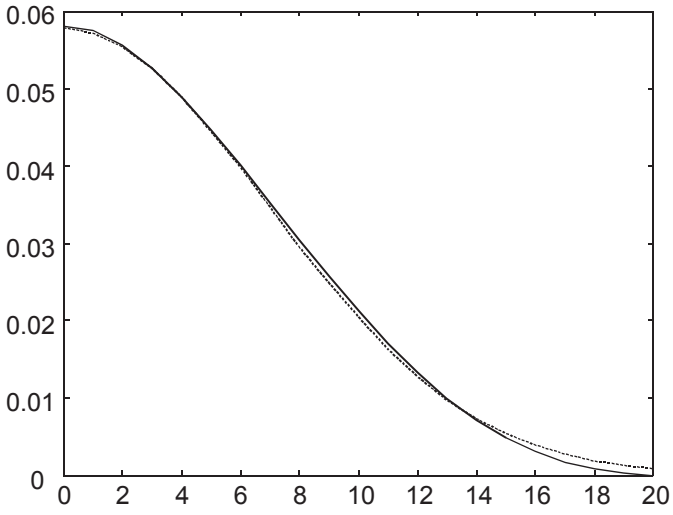

FIG. 6. Demi réponse impulsionnelle du filtre gaussien et son approximation par POAG en pointillé.

B. Equations du filtre POAG

Un peu rébarbatif mais allons y. A chaque valeur de W correspond un filtre de réponse impulsionnelle $h_W(n)$ non nulle sur l'intervalle $[-W, +W]$.

$$h_W(n) = (w+2-|n|)(w+1-|n|)(-3n^2+(2w+3)\,|n|+w(w+3))$$

c'est donc bien un polynôme en n de degré 4.

La réalisation de ce filtre transverse nécessite (compte-tenu de la symétrie des coefficients) $w+1$ multiplications et $2w$ additions. Nous allons montrer ci-dessous que sa réalisation sous forme récursive (avec 5 intégrateurs en casacade) conduit à un nombre de calculs indépendant de w : 4 multiplications et 10 additions.

Quand la réponse impulsionnelle est un polynôme de degré N, on peut toujours (au moins avec un logiciel de calcul symbolique) mettre la fonction de transfert en r sous la forme d'une fraction rationnelle en r possédant $N+1$ pôles en $r = 1$ et bien sûr des zéros en nombre identiques pour les compenser. Ainsi :

$$H_W(r) = \frac{A_W(r)}{(1-r)^5}$$

avec [4] :

$$\begin{aligned} A_W(r) \quad = \quad & [w(r^{-(w+2)} - r^{(w+2)}) \\ & +(w+3)(r^{(w+1)} - r^{-(w+1)}) \\ & +(2w+3)(r^{-1} - r)] \cdot r^2(1+r) \end{aligned}$$

Il résulte de la présence de pôles sur le cercle unité que la stabilité n'est assurée qu'à condition qu'aucune erreur de calcul ne soit commise.

- les coefficients du polynôme en r : $A_W(r)$ doivent être entiers ;
- les calculs ne peuvent être faits en flottant [5] (problème de précision).

[4] à une constante d'amplification près

[5] Tout comme les offsets pour l'intégrateur analogique, les erreurs dans les additions (inévitables en flottant) font diverger les intégrateurs numériques

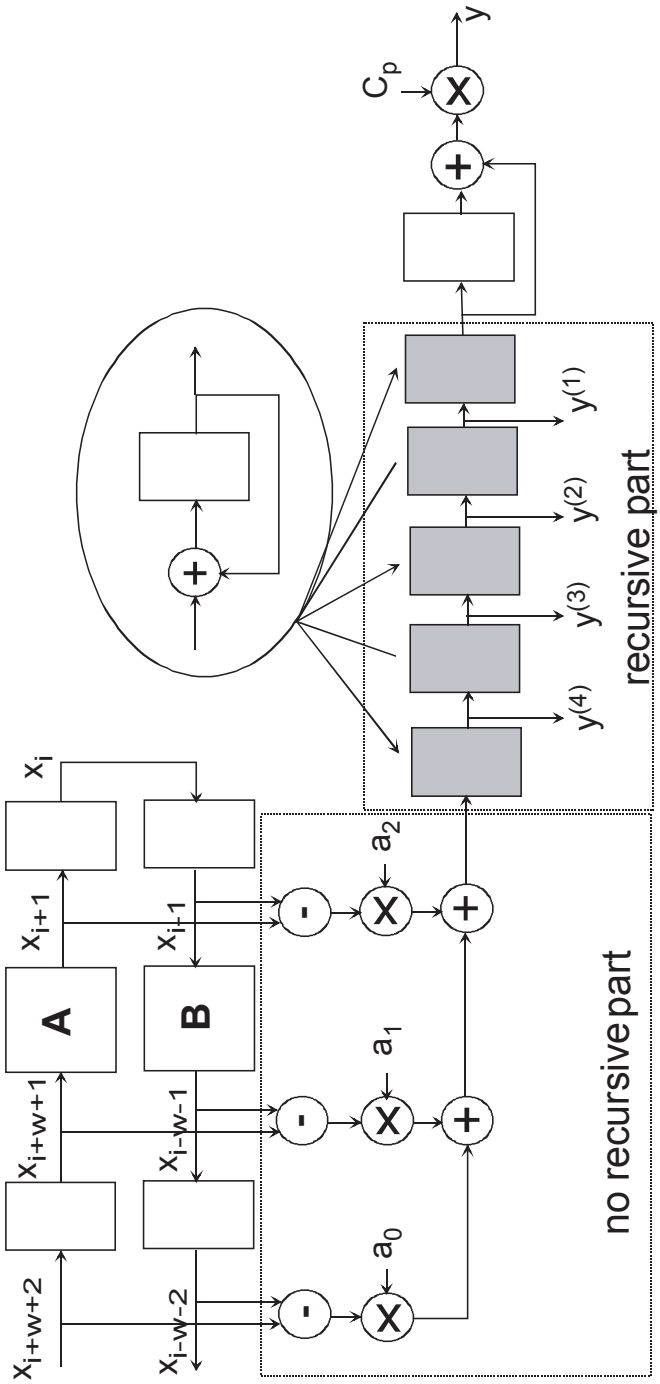

FIG. 7. Schéma fonctionnel de réalisation du filtre POAG.

– les calculs se font en complément à deux sans approximation (problème de dépassement de capacité de représentation).

C. Réalisation du filtre POAG

La figure 7 montre l'organisation fonctionnelle de la réalisation du filtre POAG qui se déduit simplement de son équation. On en dérive facilement une implantation matérielle en FPGA ou bien logicielle. L'algorithme de calcul correspondant est précisé ci-dessous dans le cas d'une largeur de filtre W quelconque. Pour alléger, on pose :

$a_0 = w,\ a_1 = w+3, a_2 = 2w+3.$

$$
\begin{aligned}
n &= a_0(x_{i+w+2} - x_{i-w-2}) + a_1(x_{i-w-1} - x_{i+w+1}) \\
&\quad + a_2(x_{i+1} - x_{i-1}) \\
t_1 &= t_1 + n \\
t_2 &= t_2 + t_1 \\
t_3 &= t_3 + t_2 \\
t_4 &= t_4 + t_3 \\
t_5 &= t_5 + t_4 \\
y_i &= C'_p(t_6 + t_5) \\
t_6 &= t_5
\end{aligned}
$$

La validation de cet algorithme constitue un bon exercice ainsi que son implantation dans un cas particulier de valeur de W (ne pas oublier l'initialisation des différentes variables intermédiaires).

V. ORIGINE "RECHERCHE" DE CE TRAVAIL

Bouclons la boucle. Comment a t'on découvert le filtre POAG ? Un travail théorique à la suite de Canny sur la définition de critères de qualité (voir figure 8) pour les filtres *numériques* détecteurs de contours a permis l'établissement des équations d'une classe de filtres optimaux vis-à-vis de ces critères [1]. En observant le degré d'optimalité de cette classe de filtres, il est apparu qu'un filtre existait pour des valeurs asymptotiques des paramètres qui n'était que légèrement sous-optimal mais possédait l'originalité d'avoir une réponse impulsionnelle polynômiale : POAG. La fonction de transfert possède donc des pôles sur le cercle unité. Pour qui a déjà implanté en logiciel ou en matériel les filtres récursifs de Shen et de Deriche [6] nécessitant à cause de leur partie causale et anti-causale un double balayage par ligne, l'attrait de pôles sur le cercle unité (là où se rejoignent causal et anti-causal) était évident : réduction du nombre de calculs, du nombre de balayages, du nombre de mémoires. La ressemblance de ces filtres avec le filtre gaussien avait déjà été soulignée par Canny. Parce que le filtre gaussien est largement utilisé et peut se révéler gourmand en calculs, il semblait alors opportun de lui substituer POAG [7], [5]. Et la mise en œuvre concrète de ce filtre a réservé bien des surprises à l'auteur : divergence en calcul flottant, gestion des saturations ...

VI. RÉCEPTION PAR LES ÉTUDIANTS, CONCLUSION

Cette étude du filtre POAG n'a pas donné lieu à un TP mais à une partie de mini-projet portant sur la détection de contours (1 binôme). Les 24 étudiants travaillaient sur 12 mini-projets différents. Trois binômes ont été déçus de ne pas aboutir. Deux binômes ont vécu ce travail de façon relativement neutre. Cinq on jugé très positivement leur mini-projet parlant même de réconciliation avec le traitement du signal. Enfin les deux binômes qui avaient des sujets liés à la détection de contours étaient tellement "imprégnés" qu'ils n'avaient en bilan que des propositions d'amélioration des systèmes qu'ils avaient produit !

Si le programme de licence ne contient que des fondamentaux, alors, par définition, ceux-ci sont exploités dans la

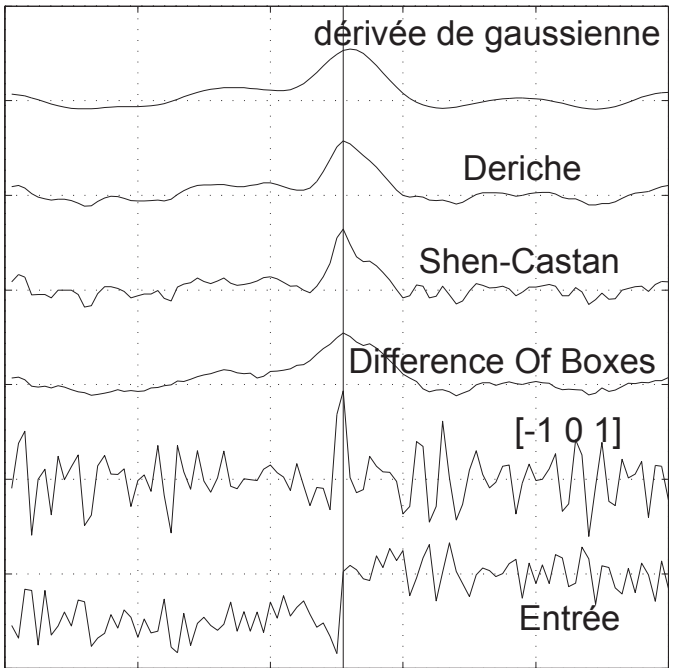

FIG. 8. On cherche à détecter la transition franche au centre du signal **Entrée** qui est entaché de bruit. Pour cela, on applique un filtre passe-bas de lissage puis on soustrait au résultat le résultat du pixel voisin de gauche (ce qui réalise une dérivation discrète). Ces deux opérations : lissage et dérivation sont combinées par chacun des filtres présentés ((1) filtre gaussien, (2) filtre de Deriche : lissage récursif du 2nd ordre appliqué de gauche à droite et de droite à gauche, (3) filtre de Shen : lissage récursif du 1er ordre appliqué de gauche à droite et de droite à gauche, (4) Difference Of Boxes : filtre transverse de taille $2w+1$ avec w coeffs à -1 à gauche, 0 au centre et w coeffs à +1 à droite). Les maxima de la sortie des filtres sont situés au centre de la transition. On remarque des comportements différents suivant les filtres appliqués. Certains détectent beaucoup de maxima dus au bruit ([-1 0 1], Shen), certains délocalisent la transition centrale (dérivée de gaussienne), certains offrent un bon contraste (différence de niveau) entre maximum vrai et maxima dus au bruit facilitant ainsi la réalisation d'un seuillage pour éliminer les mauvais maxima.

pratique des chercheurs et des ingénieurs. Il y a grand intérêt à ce que ceux-ci tentent d'extraire de leurs pratiques, de leurs expériences des supports à la pédagogie qui soient exploités avec les étudiants dès les premières années de licence.

Par ailleurs, j'ai pu constater que ce n'était pas les maths que rejettent les étudiants mais les calculs compliqués. Vive les exemples simples, sans calculs et qui ne servent à rien ! Ou presque !

ANNEXE A : PROGRAMME MATLAB DE DÉTECTION DE CONTOUR PAR LE FILTRE POAG

```
function contour(input,w)
  (ex : contour('gris.bmp',5) )
image=imread(input); lecture image
image=double(image); en flottant
image=(0.3*image(:,:,1)+0.59*image(:,:,2)
+0.11*image(:,:,3));
```

image couleur transformée en image niveaux de gris : image=0.3xRouge+0.59xVert+0.11xBleu

Calcul du vecteur h des coefficients du filtre

```
k=-w:w;
Cp=5/(2*w)/(w+1)/(w+2)/(w+3)/((2*w)+3);
h=Cp.*(w+2-abs(k)).*(w+1-abs(k))
  .*[(-3.*k.*k)+(2.*w+3).*abs(k)+w.*(w+3)]
[L,K]=size(image)
for l=1:L Lissage horizontal
  image_lh(l,:)=filter(h,1,image(l,:));
end
for k=1:K Lissage vertical
  image_ll(:,k)=filter(h,1,image_lh(:,k));
end Sauvegarde de l'image lissée
  imwrite(uint8(image_ll),'flou.bmp','bmp');
  Calcul des filtres de dérivation
rhh=[1 -1]; rhv=[1 1];
rvh=[1 1]; rvv=[1 -1];
for l=1:L traitements horizontaux de dérivation
  image_dh(l,:)=filter(rvh,1,image_ll(l,:));
  image_dv(l,:)=filter(rvv,1,image_ll(l,:));
end
for k=1:K traitements verticaux de dérivation
  image_dh(:,k)=filter(rhh,1,image_dh(:,k));
  image_dv(:,k)=filter(rhv,1,image_dv(:,k));
end
  Calcul du module du gradient
image_d=(sqrt(image_dh.*image_dh
 + image_dv.*image_dv))/ (2.*sqrt(2));
Sauvegarde de l'image de gradient
imwrite(uint8(image_d),'contour.bmp','bmp');
```

RÉFÉRENCES

[1] D. Demigny, "On optimal linear filtering for edge detection," *IEEE Transactions on Image processing*, vol. 11, no. 7, pp. 728–737, july 2002.

[2] J. Canny, "A computational approch to edge detection," *IEEE Transactions on Pattern Analysis And Machine Intelligence*, vol. PAMI-8, pp. 679 – 714, 1986.

[3] R. Deriche, "Fast algorithm for low-level vision," *IEEE Transactions on Pattern Analysis And Machine Intelligence*, vol. 12, no. 1, January 1990.

[4] D. Demigny. (2008) Blog professionnel de didier demigny. [Online]. Available : http ://blogperso.univ-rennes1.fr/didier.demigny/

[5] D. Demigny, J. Pons, N. Boudouani, and lounis kessal, "Réalisation récursive temps réel de filtre rif : filtre de canny, filtre gaussien et ses dérivées," in *18 ème Colloque sur le traitement de images et du signal*. Toulouse : GRETSI, septembre 2001.

[6] M. Akil, E. Belhaire, E. B. Bourennane, Y. Berviller, D. Demigny, P. Garda, L. Kessal, L. Lacassagne, F. Lohier, M. Paindavoine, M. Robert, Y. Sorel, L. Torres, and S. Weber, *Méthodes et architectures pour le TSI en temps réel, Traité Information, Commande et Communication.* Hermès, 2001, d. Demigny coordonnateur.

[7] D. Demigny, L. Kessal, and julien Pons, "Fast recursive implementation of the gaussian filter," in *11th IFIP International Conference on Very Large Scale Integration*, no. 1. Montpellier, France : LIRMM, december 2001, pp. 339–346.