Détection de primitives linéaires et circulaires par une approche a contrario

Viorica Pătrăucean, Pierre Gurdjos, Géraldine Morin, Jean Conter

Détection de primitives linéaires et circulaires par une approche *a contrario*

Viorica Pătrăucean[1,2] Pierre Gurdjos[1] Géraldine Morin[1] Jean Conter[1]

[1]Université de Toulouse (Institut de Recherche en Informatique de Toulouse)
[2]Académie Technique Militaire de Bucarest
{vpatrauc, pgurdjos, gmorin, conter}@enseeiht.fr

Résumé

La compréhension bas niveau d'une image exige l'usage des différents détecteurs de primitives de base, telles que des segments de droite ou arcs de cercles. La plupart des détecteurs existants se confrontent à des problèmes qui ont été (et sont toujours) considérablement étudiés comme le réglage de paramètres, le contrôle du nombre de fausses détections ou le temps d'exécution. Dans cet article, nous nous intéressons à la détection à la fois des droites et des cercles dans une image, tout en contrôlant le nombre de fausses détections et sans réglage particulier de paramètres. Nous présentons un détecteur qui étend l'algorithme LSD (Line Segment Detector) aux cercles, les deux étant fondés sur une approche a contrario. Du fait que le détecteur proposé vise deux types différents de primitives, la validation a contrario est utilisée comme méthode de sélection du modèle, ce qui représente une nouveauté dans les travaux fondés sur l'approche a contrario. De plus, nous proposons une nouvelle méthode d'estimation algébrique d'un cercle, qui profite également de la direction du gradient des points contour, et non pas uniquement de la position de ceux-ci.

Mots Clef

primitive de base, détecteur a contrario, estimation algébrique.

Abstract

Low-level image understanding requires the use of different detectors for basic primitives, such as line segments or circular arcs. Most of the existent detectors deal with problems that have been (and still are) extensively studied like parameter tunning, control of number of false detections or execution time. In this paper, we focus on detecting simultaneously lines and circles in an image, while controlling the number of false detections and without any need of parameter tunning. We present an algorithm which extends the Line Segment Detector (LSD) for circles, both being based on the a contrario approach. Due to the fact that the proposed detector targets two different types of primitives, the a contrario validation is used as a criterion for model selection, which is a novelty in the a contrario-based works. In addition, we propose a new algebraic method for estimating a circle, which benefits equally from the direction of the gradient of the contour points, and not only from their position.

Keywords

basic primitive, a contrario detector, algebraic estimation.

1 Introduction

L'interprétation d'une image par une machine, objet d'étude de la *vision par ordinateur*, a comme point de départ l'identification des formes élémentaires (droites, courbes) contenues dans l'image. Dans ce but, de nombreux détecteurs ont été proposés dans la littérature. Toutefois, la majorité des algorithmes se confrontent à de multiples défis, tels que le réglage de paramètres ou le contrôle de fausses détections. De plus, la plupart du temps les détecteurs sont spécialisés : ils prennent en compte soit des droites [16, 15], soit des courbes de différents degrés [14, 22]. Vu qu'en général les images, notamment les images naturelles, ont un contenu complexe en termes de formes géométriques, plusieurs passes sont nécessaires pour accomplir une compréhension bas niveau de l'image, ce qui rend le temps de traitement assez conséquent. Dans ces travaux, nous nous intéressons aux algorithmes capables de produire en sortie l'ensemble de primitives linéaires et circulaires apparaissant dans une image, qui ne nécessite pas de réglage de paramètres et s'exécute en un temps de calcul raisonnable.

Parmi les quelques algorithmes capables de détecter simultanément des droites et des courbes (cercles) nous mentionnons ceux proposés dans [21] et [7]. Le détecteur de Taubin [21] identifie des courbes de n'importe quel degré dans une image, mais il s'avère très complexe et coûteux en temps d'exécution. De plus, les résultats ne sont pas satisfaisants pour les images bruitées. Le détecteur proposé par Etemadi [7], bien qu'il n'ait pas besoin d'un réglage particulier de paramètres, génère des nombreuses fausses détections dans le cas de textures répétitives.

En général, le contrôle du nombre de fausses détections est fortement lié au réglage de paramètres. Établir les seuils de détectabilité est le problème majeur de la plupart des

détecteurs classiques (par exemple, les différents versions de Hough qui existent pour différents types de primitives [16, 15, 14]). L'approche *a contrario*, publiée en 2000 par Désolneux *et al* [6] dans le but de formaliser les principes de l'école gestaltiste, vise notamment cet aspect. Plusieurs algorithmes se basant sur cette approche ont été proposés pour s'attaquer aux différents points essentiels dans le domaine de la vision par ordinateur : détecteurs de primitives linéaires [10], segmentation [4], calcul de la matrice fondamentale [17]. Ce sont des méthodes applicables à tout type d'images (géométriques, industrielles, naturelles), sans nécessiter un réglage préalable de paramètres. Le détecteur que nous proposons profite de cet avantage.

Par la suite nous présentons les principes fondamentaux de l'approche *a contrario* et leur application dans notre détecteur (section 2). Dans la section 3 nous détaillons une nouvelle méthode d'estimation algébrique d'un cercle, utilisée par l'algorithme de détection. Les sections 4 et 5 présentent les résultats expérimentaux et les conclusions.

2 Détection de primitives linéaires et circulaires

En général, les algorithmes de détection, pour n'importe quel type de forme, comportent deux parties essentielles distinctes : la sélection des candidats et la validation. Évidemment, dans la première phase il est important de ne pas avoir de *faux négatifs* (un objet est présent, mais on ne le détecte pas). Ensuite, selon les besoins de l'application, la méthode de validation doit réduire au minimum soit le nombre de *faux positifs* (un objet n'est pas présent, mais l'algorithme en détecte un), soit celui de faux négatifs. Bien que pour la première partie il existe des techniques largement acceptées (e.g. la transformée de Hough [2]), la partie de validation reste un enjeu, puisqu'elle dépend souvent de différents seuils, qui ne sont plus valides dès que l'image ou le type d'image change.

Une méthode efficace de validation est proposée dans [6], et elle est fondée sur l'approche *a contrario*.

2.1 Validation *a contrario* pour les segments de droites

L'approche *a contrario* [6] essaie de formuler une *théorie de la perception* au bénéfice de la vision par ordinateur, *i.e.*, de donner le support mathématique qui permet de prédire les perceptions associées (engendrées) à une image numérique. Dans ce but, elle s'inspire des lois gestaltistes pour décrire ce qu'il est *observable* comme forme dans une image, et utilise le principe de Helmholtz pour établir *des seuils généraux de détectabilité*. Ces démarches sont nécessaires pour aboutir à des algorithmes de détection de formes, libres de tout réglage spécifique de paramètres. Par la suite, nous détaillons le principe *a contrario* pour le détecteur d'alignements dans une image [6, p. 65]. Nous utilisons la même technique de validation pour notre détecteur.

Pour valider un alignement dans une image, l'approche *a contrario* donne un sens mathématique au principe de Helmholtz qui dit, de manière informelle, qu' *il n'y a pas de perception dans du bruit blanc*.

En d'autres termes, tout alignement de l'image à analyser $\mathcal{I}$ que l'on peut espérer d'observer dans une image de bruit blanc de même taille, ne produit pas une perception (n'attire pas l'attention), et donc ne devrait pas être considéré comme détection valide, puisqu'il est dû au hasard.

Par *image de bruit blanc* on désigne une image dont les valeurs des niveaux de gris sont des variables aléatoires indépendantes qui suivent une loi normale [1]. On appelle cela *l'hypothèse a contrario* H_0.

On définit la notion d'alignement de points en tenant compte de la direction du gradient en tout point, sachant qu'elle est théoriquement perpendiculaire au contour. Deux pixels d'un même *segment* [1] sont considérés alignés si les directions des gradients g_1 et g_2 associés coïncident à une certaine précision $\delta \in [0, 1]$, c.-à-d. si l'angle non orienté $\widehat{g_1 g_2} \in [0, \pi]$ vérifie $\widehat{g_1 g_2} \leq \delta\pi$. On appelle alignement de type $k(l, n, \delta)$ tout segment k de longueur l, contenant n pixels alignés à une précision δ. Puisque, sous l'hypothèse H_0, la direction du gradient en chaque point est une variable aléatoire indépendante et uniformément distribuée sur $[0, \pi]$ [6, p. 67], on en déduit que la probabilité d'avoir au moins n pixels alignés parmi l, c.-à-d. *la probabilité d'un alignement de type* $k(l, n, \delta)$ *dû au hasard*, suit une loi binomiale :

$$\mathcal{B}(l, n, \delta) = \sum_{i=n}^{l} \binom{l}{i} \delta^i (1 - \delta)^{l-i}. \tag{1}$$

Un alignement de type $k(l, n, \delta)$ sera dit ϵ-significatif si sa probabilité d'être dû au hasard est inférieure à un certain seuil, c.-à-d. vérifie

$$\mathcal{B}(l, n, \delta) \leq \epsilon / N_t \tag{2}$$

où N_t désigne le nombre total de tests (*«un alignement est-il dû au hasard ou pas ?»*) qu'on peut mener dans l'image. Il a été montré [6] que si les alignements considérés comme valides sont ceux ϵ-significatifs alors, sous l'hypothèse H_0, l'espérance du nombre d'alignements ϵ-significatifs, que l'on notera $\mathbb{E}_{H_0}$, vérifie $\mathbb{E}_{H_0} \leq \epsilon$. En conséquence, il suffit de se donner la quantité moyenne de fausses détections ϵ que l'on accepte sous l'hypothèse H_0 et on peut affirmer qu'il n'y a pas d'autre seuil de détection à fixer quand on se place dans un cadre *a contrario*.

Comme démontré dans [6, p. 76], la dépendance de la précision des résultats par rapport à la valeur de ϵ est logarithmique, donc faible, et on peut choisir de fixer la valeur $\epsilon = 1$. Avec ce choix on assume le risque d'avoir au pire *une* fausse détection, c.-à-d. d'accepter comme détection valide un alignement qui soit dû au hasard. Cette valeur s'est avérée être satisfaisante pour la majorité des applications pratiques.

1. On parle ici de segment discret, c.-à-d. d'ensemble de pixels *connexe* et *fin*.

En revenant à l'équation (2), on définit le *nombre de fausses alarmes (NFA)* d'un alignement k comme

$$NFA(k) = N_t \mathcal{B}(l, n, \delta). \quad (3)$$

Par conséquent, un segment est ϵ-significatif si $NFA(k) \leq \epsilon$. Ce test détermine si un alignement est valide ou s'il est dû au hasard.

La méthode de validation *a contrario* est globalement très efficace. Toutefois, le détecteur de segments présenté dans [6] a un inconvénient majeur : la phase de sélection de candidats est exhaustive. Tous les *segments de support* possibles que l'on peut avoir dans une image (N^4 tests puisque toute paire de points de l'image peut représenter les bouts d'un segment) sont proposé à l'étape de validation. Des opérations supplémentaires suppriment les redondances (par exemple, la suppression des faisceaux de droites obtenus sur des contours flous). Par conséquent, le temps d'exécution est très élevé. Grompone *et al.* proposent un algorithme amélioré (*Line Segment Detector ou LSD*) [10] qui utilise une validation *a contrario*.

2.2 L'algorithme *LSD*

L'algorithme LSD [10] met en valeur deux contributions importantes pour obtenir un détecteur optimal de droites. Dans un premier temps, LSD profite de l'idée de Burns *et al.* [3] pour trouver les candidats potentiels pour des segments, i.e. un segment est considéré comme une région rectangulaire (*rectangle de support*) couvrant un groupe de pixels connexes, alignés à une précision δ. Ainsi, en localisant ces groupements de pixels, il évite de tester toutes les possibilités comme dans [6]. Au final, les candidats sont acceptés ou rejetés en utilisant une validation de type *a contrario*.

Un rectangle possède cinq degrés de liberté : les coordonnées du centre de gravité (2 *ddl*), la longueur (1 *ddl*), la largeur (1 *ddl*), et l'orientation (1 *ddl*). De ce fait, le nombre de tests est $N_t \sim N^5$. Mais le calcul effectif n'est fait que pour les rectangles de support qui existent effectivement dans l'image.

Un segment candidat est accepté si

$$NFA_{segment} = N^5 \mathcal{B}(l, n, \delta) \leq 1 \quad (4)$$

où l représente le nombre total de pixels dans le rectangle et n est le nombre de pixels alignés avec la direction du premier axe d'inertie du rectangle à une précision δ.

Cet algorithme est très performant en termes de précision et de temps de calcul. En effet, en utilisant la technique de Burns *et al.* [3], la complexité devient linéaire par rapport au nombre de pixels de l'image.

Bien que toute forme géométrique puisse être approximée par des segments de droite et que le *LSD* puisse théoriquement couvrir la détection, nous avons étendu l'algorithme LSD, pour détecter à la fois des segments de droite et des arcs circulaires : *Circle and Line Segment Detector (CLSD)*. Notre détecteur s'avère nécessaire dans le cas des cercles petits, où le LSD risque de considérer les segments qui l'approximent comme des candidats non significatifs. De plus, pour les applications réelles, il est plus pratique d'avoir un détecteur pour les deux types de primitives (au lieu de mettre en place par exemple des techniques qui estiment un cercle à partir de segments).

2.3 Validation *a contrario* pour les arcs de cercles

Notre contribution est de montrer que le raisonnement pour la validation *a contrario* des arcs circulaires reste identique à celui utilisé dans *LSD*, avec quelques adaptations spécifiques. Les *rectangles de support* sont remplacés par des arcs de couronnes circulaires, appelés *couronnes de support*. Un arc de couronne circulaire est défini par six paramètres : le centre (2 *ddl*), le rayon (1 *ddl*), deux angles qui délimitent l'arc (2 *ddl*) et la largeur de la couronne (1 *ddl*) ; le nombre de tests à considérer est donc $N_t \sim N^6$. Un arc candidat est accepté si

$$NFA_{arc} = N^6 \mathcal{B}(l, n, \delta) \leq 1 \quad (5)$$

où l représente le nombre total de pixels dans la couronne de support et n est le nombre de pixels dont les directions du gradient coïncident à une précision δ avec les directions des droites passant par les pixels et le centre du cercle estimé.

2.4 Algorithme général

Pour implémenter notre détecteur (*CLSD*) nous avons étendu l'algorithme *LSD* avec les fonctions nécessaires pour traiter les arcs circulaires. Nous présentons dans le pseudocode Algorithme 1 les étapes principales de l'algorithme. Pour aider à la compréhension, nous expliquons succinctement les fonctions de *LSD* et les fonctions rajoutées, entourées dans Algorithme 1.

En premier lieu, nous faisons le glossaire des paramètres qui apparaissent dans l'algorithme : le seuil ρ (seuil minimum sur la norme du gradient pour éviter d'analyser les pixels dont la direction du gradient aurait pu être altérée par le bruit de quantification [10]) et la précision δ peuvent être considérés comme des paramètres internes et fixés une fois pour toutes. Les valeurs utilisées sont : $\rho \sim 5$, $\delta = 0.125$ (cette dernière valeur a été obtenue empiriquement sur les tests effectués ; elle a été utilisée aussi dans les travaux similaires [10], [3]). Le gradient est calculé sur des fenêtres de 2×2 pixels. Cf. section précédente, on considère $\epsilon = 1$.

CLSD démarre avec le calcul des orientations et des normes des gradients ; une liste des pixels triés selon la norme du gradient est calculée aussi, afin que les pixels avec un gradient «fort» (susceptibles d'être sur un contour) soient traités en premier. Nous mettons à jour systématiquement un tableau avec l'état des pixels (*« utilisé »* ou *« non utilisé »*) qui ne contient au début que des pixels non traités.

Ensuite, tout pixel non utilisé dans la liste triée, sert comme germe pour construire un *rectangle de support* (fonction *AugmenteRegion*), c.-à-d. nous cherchons parmi les voisins du germe ceux qui sont alignés à une précision δ. Le calcul

Algorithme 1 CLSD – Circle and Line Segment Detector

ENTRÉES: image I, paramètres ρ, δ, ϵ.
SORTIES: une liste $outrect$ de segments et une liste $outcerc$ d'arcs de cercle.

```
(Angles, MagGrad, ListTri) ← Grad(I, ρ) ;
Status(touspixels) ← PasUtilis ;
pour tout pixel P dans ListTri faire
  si Status(P) = NonUtilis alors
    region ← AugmenteRégion(P, δ, Status) ;
    rect ← ApproxRect(region) ;
    nfasegment ← NFAsegment(rect) ;
    répéter
      cercestim ← EstimCercle(region) ;
      region ← AugmenteCercle(region, cercestim, δ,
      Status) ;
    jusqu'à il n'y a plus de pixels à rajouter ;
    cerc ← ApproxCerc(region) ;
    nfacercle ← NFAcercle(cerc) ;
    si nfacercle < nfasegment alors
      si nfacercle < ϵ alors
        Ajout cercestim à outcerc ;
        Status(regcerc) ← Utilis ;
      finsi
    sinon si nfasegment < ϵ alors
      Ajout rect à outrect ;
      Status(rect) ← Utilis ;
    sinon
      Status(regcerc, rect) = NonUtilis ;
    finsi
  finsi
fin pour
```

se poursuit récursivement en cherchant les voisins alignés aux nouveaux pixels rajoutés, jusqu'à ce qu'aucun pixel ne puisse être rajouté. Le rectangle obtenu à la fin de cette étape représente en fait un segment de longueur égale à la longueur du rectangle et de direction parallèle avec le premier axe d'inertie du rectangle.

Nous étudions maintenant la possibilité que ce groupement de pixels alignés soit en fait situé sur un arc de cercle. Dans ce but, nous estimons un cercle à partir de ces points (fonction *EstimCercle*), en utilisant une méthode algébrique peu coûteuse qui est détaillée dans la section suivante. Une nouvelle fonction *AugmenteCercle* est lancée, avec le même but que la fonction *AugmenteRegion*, c.-à-d. trouver tous les voisins dont les directions du gradient coïncident à une précision δ avec la direction des droites passant par les voisins analysés et le centre du cercle. Quand aucun pixel ne peut plus être rajouté, la fonction *ApproxCerc* calcule l'arc de couronne qui couvre les pixels sélectionnés.

Une fois qu'on possède le *rectangle de support* et la *couronne de support*, la signifiance des deux est calculée avec les formules (4) et (5). Celui qui a la valeur de *NFA* la plus petite est le plus significatif, donc gardé comme détection valide (segment de droite ou arc de cercle).

Cette dernière étape mérite un commentaire sur l'évolution de l'usage du NFA. Dans les premiers travaux sur l'approche *a contrario*, la valeur du NFA était utilisée comme critère de rejet d'un candidat non significatif [6]. Par la suite elle a servi également comme moyen de comparer la signifiance de deux événements appartenant à la même classe (e.g., le choix du segment le plus significatif parmi plusieurs segments candidats [12, 6]). Dans le cadre de notre algorithme, nous introduisons, de manière rudimentaire, une troisième utilisation possible du NFA, évoquée dans [6, p. 246], notamment dans le problème de sélection du modèle. En effet, étant donné que l'algorithme CLSD vise la détection à la fois des primitives linéaires et circulaires, nous sommes amenés en face d'un problème de sélection entre deux interprétations possibles, appartenant à deux classes différentes. Dans ce cas, nous utilisons la valeur du NFA comme critère de sélection binaire de modèle : selon la valeur du nombre de fausses alarmes, nous prenons la décision sur le modèle (arc de cercle ou segment de droite) qui approxime le mieux les données fournies.

3 Nouvelle méthode algébrique d'estimation d'un cercle

L'estimation d'une primitive à partir de mesures effectuées dans les images est un problème récurrent dans le domaine de la vision par ordinateur. Nous restreignons la discussion à l'estimation de primitives circulaires et nous distinguons deux classes de méthodes d'estimation : *géométriques* et *algébriques*. L'estimation géométrique [9] réside dans la minimisation de la somme des carrés de distances euclidiennes entre les points et le cercle estimé. Étant donnés n points $p_i = (x_i, y_i), 1 \leq i \leq n$, la fonction objectif est

$$\mathcal{F} = \sum_{i=1}^{n} d_i^2, \tag{6}$$

avec $d_i = \sqrt{(x_i - a)^2 + (y_i - b)^2} - R$, pour un cercle $\mathcal{C}$ de centre $O(a, b)$ et de rayon R. Cette approche est non linéaire en (a, b, R) les paramètres à estimer, et la plupart des méthodes proposées utilisent des procédés itératifs, donc coûteux, pour arriver à la solution. En revanche, les méthodes algébriques [19] substituent une distance algébrique à la distance géométrique, rendant le problème linéaire. En effet, on considère le cercle $\mathcal{C}$ représenté par son équation homogène matricielle :

$$\tilde{p}^T C \tilde{p} = 0, \tag{7}$$

où

$$C = \begin{pmatrix} 1 & 0 & -a \\ 0 & 1 & -b \\ -a & -b & a^2 + b^2 - R^2 \end{pmatrix} \tag{8}$$

est la matrice du cercle, p le vecteur $(x, y)^T$ des coordonnées cartésiennes d'un point et $\tilde{p}$ le vecteur augmenté $\tilde{p} = (p^T, 1)^T$. Le problème est :

$$\min_{u, u \neq 0} \|Du\|^2, \tag{9}$$

où $D = \begin{pmatrix} 1 & 2p_1^T & p_1^T p_1 \\ \vdots & \vdots & \vdots \\ 1 & 2p_n^T & p_n^T p_n \end{pmatrix}$; u est le vecteur de paramètres correspondant aux éléments non nuls de la matrice C. Sous la contrainte $\|u\| = 1$, il s'agit d'un problème de moindres carrés totaux où l'on minimise la somme des puissances des points au cercle. La solution est donnée par le vecteur propre correspondant à la plus petite valeur propre de la matrice $D^T D$.

Pour le détecteur décrit dans la section précédente, nous avons besoin d'une méthode d'estimation d'un cercle robuste aux données incomplètes et, en même temps, peu coûteuse (non-itérative). Afin de répondre à ces exigences, nous proposons une méthode d'estimation algébrique qui utilise efficacement toute l'information disponible dans une image.

La plupart des méthodes algébriques existantes utilisent exclusivement la position des points donnés pour estimer le cercle, en résolvant l'équation (9). Il existe toutefois peu de méthodes qui essaient d'améliorer les performances en introduisant une contrainte sur la direction du gradient. Nous résumons les principes de ces méthodes parce que notre approche se situe dans cette catégorie.

L'opérateur proposé par Forstner *et al.* [8] calcule le centre d'un cercle comme le point d'intersection (au sens de moindres carrés) des droites dont les vecteurs directeurs sont parallèles au gradient des pixels situés sur le contour du cercle. Guennebaud *et al.* [13] reprennent l'idée dans le cas 3D, pour estimer le centre d'une sphère. De plus, ils utilisent par la suite la position des points pour calculer le rayon. Puisque cette idée n'est pas applicable aux ellipses (les droites passant par les points et dirigées par le vecteur gradient ne concourent pas en le centre de l'ellipse), Ouellet *et al.* [18] passent dans l'espace dual et utilisent la relation pôle – droite polaire

$$\tilde{p} = C^* l, \tag{10}$$

où C^* est la conique duale associée à une conique C ($C^* \sim C^{-1}$) et le point $\tilde{p} \sim (x, y, 1)^T$ est le pôle de la droite l par rapport à C. Si $\tilde{p}$ appartient à C, sa droite polaire est tangente à C en $\tilde{p}$, et elle peut être obtenue en calculant l'équation de la droite perpendiculaire au gradient en ce point :

$$l = \tilde{p} \wedge \tilde{g}, \tag{11}$$

où $\tilde{g} = (-g_y, g_x, 0)^T$ est le vecteur gradient et $\wedge$ désigne le produit vectoriel. L'ensemble de polaires obtenues pour les pixels situés sur le contour d'une ellipse permettent le calcul de l'*ellipse duale* C^* [18] par l'équation $l_i^T C^* l_i = 0$. Ainsi, en passant dans l'espace dual, l'information de tangence en un point $\tilde{p}$ (qui est plus forte que la contrainte sur la direction du gradient utilisée dans [8, 13]) est mise en valeur, mais la contrainte sur la position du point $\tilde{p}$ sur l'ellipse est perdue.

Ces méthodes, bien qu'elles améliorent les résultats de l'estimation tout en restant simples, sont «sous-optimales»

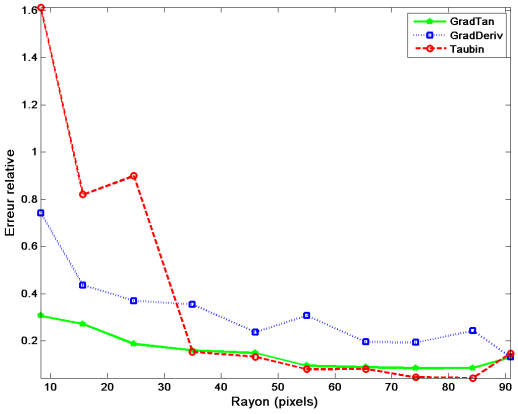

FIGURE 1 – Comparaison entre l'erreur relative sur le rayon du cercle estimé, en utilisant trois méthodes : notre méthode (GradTan), GradDer et Taubin. Les images ont été reparties dans 10 classes, selon la valeur du rayon (1ère classe : rayon de 5 - 10 pixels, 2ème classe : rayon de 10 - 20 pixels, 3ème classe : rayon de 20 - 30 pixels etc.). Pour chaque classe, le rayon moyen a été calculé, ainsi que l'erreur moyenne correspondante.

du point de vue des contraintes imposées. Nous proposons une méthode qui utilise à la fois la position des points et l'information de tangence en ces points.

Pour cela, nous considérons les deux expressions qui donnent la tangente l au cercle C : l'équation duale de (10), et l'équation (11). En les remplaçant dans l'équation $l \wedge l = 0$, nous obtenons une nouvelle contrainte linéaire en C. Avec (7), le système d'équations devient :

$$\begin{cases} \tilde{p}^T C \tilde{p} = 0 \\ (\tilde{p} \wedge \tilde{g}) \wedge C\tilde{p} = 0. \end{cases} \tag{12}$$

La solution est obtenue par décomposition en valeurs singulières.

Les performances de la méthode de Guennebaud *et al.* [13] (notée *GradDer*) et de la méthode proposée (notée *GradTan*) sont comparées en Figure 1. Une méthode d'estimation algébrique classique (i.e. qui n'utilise que la position des points) est également rajoutée à notre étude (*Taubin*). Selon [5], parmi les méthodes d'estimation algébrique, l'approche de Taubin [21] a les meilleures performances quand l'estimation n'est faite qu'en connaissant des points qui se trouvent sur un petit arc.

Les résultats (Figure 1) montrent que notre méthode est la plus adaptée pour des cercles de petits rayons (5-20 pixels). La contrainte de tangence imposée par notre méthode est plus forte que la contrainte sur la direction utilisée

Erreur moyenne (maximale) sur le centre

Méthode	0%	2%	4%	6%	8%
GradTan	0.00628(0.0337)	0.0657(0.2175)	0.0989(0.3365)	0.1217(0.3397)	0.1381(0.4030)
Ouellet	0.00556(0.0315)	0.0721(0.2256)	0.1129(0.3727)	0.1418(0/4504)	0.1592(0.4177)

Erreur moyenne (maximale) sur le rayon

Méthode	0%	2%	4%	6%	8%
GradTan	0.9646(0.1.6053)	0.9624(1.6542)	0.9488(1.6589)	0.9543(1.7351)	0.9566(1.7459)
Ouellet	1.1800(1.7557)	1.2191(1.8753)	1.2727(2.1181)	1.3410(2.6373)	1.3594(2.6373)

Tableau 1 – Erreur sur l'estimation du centre/rayon avec les méthodes *GradTan* et *Ouellet* pour 5 niveaux de bruit différents.

Erreur moyenne (maximale) sur le centre pour des données incomplètes (bruit 2%)

Méthode	75% du contour	50% du contour
GradTan	0.1136(0.4465)	0.5972(4.8615)
Ouellet	0.2018(1.0942)	3.4944(8.3421)

Erreur moyenne (maximale) sur le rayon pour des données incomplètes (bruit 2%)

Méthode	75% du contour	50% du contour
GradTan	0.5159(0.9927)	0.9068(5.1210)
Ouellet	0.8299(1.5326)	4.6549(12.0225)

Tableau 2 – Erreur sur l'estimation du centre/rayon avec les méthodes *GradTan* et *Ouellet* sur des données incomplètes

par *GradDer*. La méthode de Taubin a des performances faibles dans cette situation. Pour des cercles de rayon plus grand (> 35 pixels), la précision de la méthode de Taubin est proche de celle de notre méthode (voire la dépasse légèrement). Ceci s'explique par le fait que l'influence du même bruit sur la position des points est plus prégnante pour des rayons petits (donc la direction du gradient est plus fiable), alors que au fur et à mesure que le rayon augmente, la position devient de plus en plus fiable, donc la méthode de Taubin plus précise.

Les tests ont été menés sur 265 petites images réelles, contenant chacune un cercle imprimé sur un papier blanc. Les cercles ont des rayons compris entre 5 et 90 pixels, et les images ont été prises dans des conditions d'éclairage différentes. L'estimation est menée sur des arcs de cercle de longueur de $22,5^\circ$ la valeur en degrés de $\delta\pi$, la précision considérée dans notre détecteur *a contrario*, qui permet au processus de chainage de pixels (fonction *AugmenteRégion* dans Algorithme 1) de regrouper les pixels connexes alignés. Si ces pixels se trouvent sur le contour d'un cercle, le chainage renverra un arc de cercle de $\sim 22,5^\circ$.

Nous avons facilement adapté la méthode proposée pour estimer des ellipses, afin de comparer les résultats avec la méthode de Ouellet *et al.* [18]. Les résultats de notre méthode (Tableau 1), obtenus sur des images de synthèse, sont visiblement meilleurs, surtout dans les cas où l'estimation est menée sur des données incomplètes (Tableau 2).

4 Résultats

Nous présentons des résultats de notre détecteur, obtenus sur des images de provenance variée : images naturelles, images de formes géométriques. La figure 2 montre les résultats obtenus sur une image issue d'un projet industriel qui vise la lecture et l'identification du code à bulles [20]. Du point de vue géométrique, un code à bulles est un ensemble de cercles (approximant les bulles) et éventuellement des segments (certaines formes du code à bulles possèdent un repère utilisé pour faciliter la gestion des rotations / translations). Nous envisageons des tests extensifs pour évaluer les performances de notre détecteur en termes de précision et de temps d'exécution, en comparaison avec d'autres approches.

5 Conclusion

Cet article présente un détecteur de primitives linéaires et circulaires (*CLSD*), qui reprend les idées de base de l'algorithme *LSD* : une sélection efficace des candidats et une validation *a contrario*. Le fait d'être fondé sur l'approche *a contrario* permet un contrôle du nombre de fausses détections, qui n'exige pas de réglage de paramètres. En même temps, dans le détecteur proposé, la méthode de validation *a contrario* est utilisée d'une façon inédite : elle sert comme critère de sélection du modèle entre les deux interprétations possibles (segment de droite ou arc de cercle). De plus, une nouvelle méthode d'estimation algébrique d'un cercle est proposée. Du fait qu'elle utilise des contraintes sur la position des points contour et sur la direction du gradient en ces points, elle s'avère robuste au bruit et aux données incomplètes. Un autre avantage est la facilité d'adaptation de notre méthode pour l'estimation d'ellipses. Pour améliorer le détecteur proposé, nous songeons à une nouvelle approche pour la sélection des candidats de primitives circulaires, puisque la méthode actuelle, fondée sur une estimation algébrique, peut représenter une source d'erreurs en introduisant des faux négatifs.

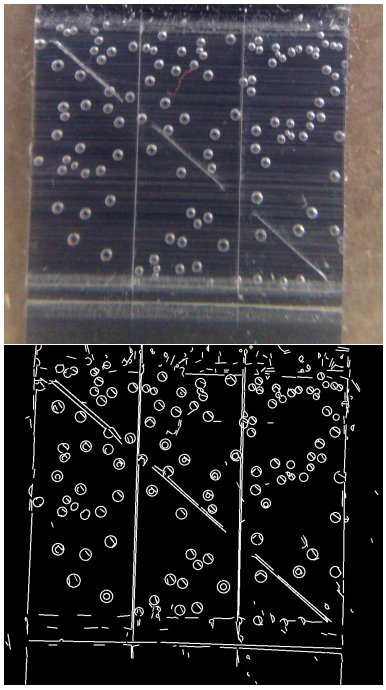

FIGURE 2 – Résultats obtenus sur une image de code à bulles.

Remerciements

Le code de LSD, utilisé par notre détecteur, est disponible en ligne sur Image Processing On Line[11]. Nous remercions Rafael Grompone von Gioi pour sa collaboration sur ce travail.

Références

[1] F. Attneave. Some informational aspects of visual perception. *Psychol Rev*, 61(3) :183–193, May 1954.

[2] D.H. Ballard. Generalizing the hough transform to detect arbitrary shapes. *Pattern Recognition*, 13(2) :111 – 122, 1981.

[3] J. B. Burns, A.R. Hanson, and E.M. Riseman. Extracting straight lines. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-8(4) :425 –455, 1986.

[4] N. Burrus, T.M. Bernard, and J.M. Jolion. Image segmentation by a contrario simulation. *Pattern Recognition*, 42(7) :1520 – 1532, 2009.

[5] N. Chernov and C. Lesort. Least squares fitting of circles. *J. Math. Imaging Vis.*, 23 :239–252, November 2005.

[6] A. Desolneux, L. Moisan, and J. M. Morel. *From Gestalt Theory to Image Analysis*, volume 34. Springer-Verlag, 2008.

[7] A. Etemadi. Robust segmentation of edge data. In *International Conference on Image Processing and its Applications, 1992*, pages 311 –314, April 1992.

[8] W. Forstner and E. Gulch. A fast operator for detection and precise location of distinct points, corners and centres of circular features. In *Proceedings of Intercommission Conference on Fast Processing of Photogrammetric Data*, pages 281 – 305, 1987.

[9] W. Gander, G.H. Golub, and R. Strebel. Least-squares fitting of circles and ellipses. *BIT*, 43 :558–578, 1994.

[10] R. Grompone von Gioi, J. Jakubowicz, J.-M. Morel, and G. Randall. Lsd : A fast line segment detector with a false detection control. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(4) :722 –732, april 2010.

[11] R. Grompone von Gioi, J. Jakubowicz, and G. Randall. Image processing on line. http ://www.ipol.im.

[12] R. Grompone von Gioi, J. Jakubowicz, and G. Randall. Multisegment detection. In *IEEE International Conference on Image Processing, 2007. ICIP 2007.*, 16 2007.

[13] G. Guennebaud and M. Gross. Algebraic point set surfaces. *ACM Trans. Graph.*, 26, July 2007.

[14] L.Y. Jiang. Fast detection of multi-circle with randomized hough transform. *Optoelectronics letters*, 5 :397–400, 2009.

[15] J. Matas, C. Galambos, and J. Kittler. Progressive probabilistic hough transform. *Transform*, 24(4) :1–10, 1998.

[16] J. Matas, C. Galambos, and J. Kittler. Robust detection of lines using the progressive probabilistic hough transform. *Computer Vision and Image Understanding*, 78(1) :119 – 137, 2000.

[17] L. Moisan and B. Stival. A probabilistic criterion to detect rigid point matches between two images and estimate the fundamental matrix. *International Journal of Computer Vision*, 57 :201–218, 2004.

[18] J.N. Ouellet and P. Hébert. Precise ellipse estimation without contour point extraction. *Machine Vision and Applications*, 2008.

[19] V. Pratt. Direct least-squares fitting of algebraic surfaces. *SIGGRAPH Comput. Graph.*, 21 :145–152, August 1987.

[20] Prooftag. Code à bulles. http ://prooftag.fr.

[21] G. Taubin. Estimation of planar curves, surfaces, and nonplanar space curves defined by implicit equations with applications to edge and range image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1115–1138, 1991.

[22] C. Xing and J. Wang. An improved method for circle detection. In *Proceedings of the First International Conference on Intelligent Robotics and Applications : Part I*, ICIRA '08, pages 463–468, Berlin, Heidelberg, 2008. Springer-Verlag.

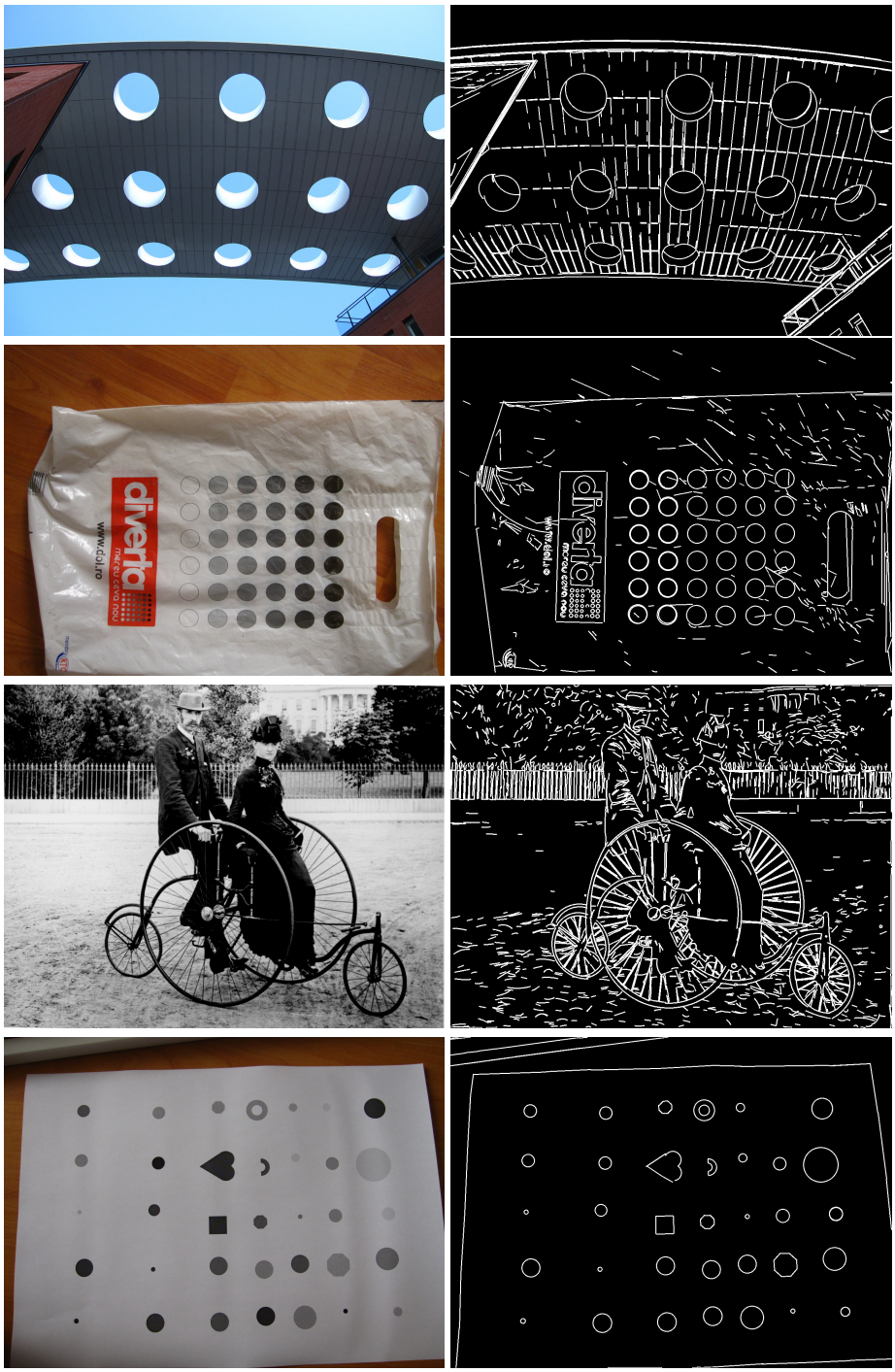

FIGURE 3 – Les résultats pour la deuxième image montrent les limitations du détecteur : la rangée de petits cercles proches du logo de la couleur la plus proche du fond manque presque complètement. En effet, la méthode d'estimation algébrique n'arrive pas à estimer un candidat valide, en raison du rayon très petit ($\sim$ 5 pixels) et du contraste faible. Sur la dernière image comportant des formes géométriques, les octogones sont bien distingués des cercles : la validation a contrario a fonctionné correctement comme critère de sélection de modèle.