Détection de visages dans des séquences vidéo à l'aide d'arbres de décision

F. Fleuret J. M. Vézien

INRIA Rocquencourt *

Domaine de Voluceau BP 105 78153 Le Chesnay Cedex
{francois.fleuret, jean-marc.vezien}@inria.fr

Résumé

Nous proposons ici une application des arbres binaires de décision à la détection d'un visage dans une séquence vidéo préalablement segmentée par champs de vitesses. L'utilisation d'un grand nombre d'arbres dont nous combinons les réponses permet d'augmenter considérablement la robustesse de la détection. Nous introduisons la notion d'instanciations des réponses des arbres binaires, notion que nous utilisons pour localiser le visage détecté.

Mots Clef

Arbres de décision, détection de visage, instanciation

Abstract

We proposed in this paper an application of binary decision trees to face detection in a video sequence which has been previously segmented using velocity fields. Using a large number of decision trees increase in a dramatic proportion the stability of this detection. We introduce the idea of instanciation of trees'answers, idea we use to localise the detected face.

Keywords

Decision trees, face detection, instanciation

1 Introduction

Les arbres de classification [2, 10, 11, 3, 5, 12] constituent un algorithme non paramétrique standard de reconnaissance avec apprentissage. Les nombreuses applications qui en ont été faites (par exemple pour la reconnaissance de caractères manuscrits [1], ou la reconnaissance d'objets rigides [6]) ont montré qu'ils constituaient un algorithme non paramétrique adapté à la reconnaissance. Nous proposons ici une application de cette technique à une tâche différente : la détection (et la localisation) d'un visage dans une séquence vidéo.
Ce problème de la détection de visages a été abordé de multiples manières différentes, par exemple à l'aide de réseaux de neurones artificiels [13, 15, 4], de centroïdes gaussiens [14], de "support vector machines" [9], ou d'estimations de densités et d'eigenfaces [7]. Notre approche démontre l'intérêt des arbres de décisions dans un tel contexte, et comment ils peuvent être utilisés efficacement sur un flux d'images.
Le pré-traitement des images consiste en une extraction de contours à l'aide d'une technique multi-échelle, avec quantification de l'orientation des bords (cf. §2). De plus, pour isoler les zones d'intérêt, les images sont segmentées par champs de vitesses (cf. §3).
L'algorithme de détection que nous proposons ici est une extension de la technique de reconnaissance par arbres de décision de [6]. La démarche consiste à opérer une reconnaissance en ne considérant que deux classes : la première contenant les images dans lesquelles se trouve un visage, et la seconde toutes les autres. Les arbres construits sur ces classes permettent donc de déterminer si une image contient ou pas un visage. Comme dans [6], cette reconnaissance se ramène à la recherche sur l'image d'un graphe dont les sommets se positionnent sur certains points caractéristiques, et dont les arêtes vérifient des contraintes géométriques (cf. § 4). Ce graphe est recalé en plusieurs positions sur l'image, et ces différentes positions, que nous appelons des "instances", sont utilisées pour localiser le visage (cf. § 5). On améliore la robustesse de cette technique en utilisant un grand nombre d'arbres dont on combine les résultats, et en tenant compte de la cohérence temporelle entre deux images successives (cf. §6).

2 Recodage initial de l'image

La détection de contours présentée dans [6] consistait en un recodage adaptatif des images. Au cours de l'ap-

*Ce projet a été financé en partie par le CTI - CNET 95 N° 95 7812 95 1082

FIG. 1 – *Images extraites de la séquence vidéo étudiée. Les quatre images du haut sont prises dans la séquence de test, les deux du bas sont des exemples des images utilisées à l'aprentissage.*

prentissage, l'algorithme construisait un arbre de classification qui permettait de recoder l'imagette 4×4 au voisinage de chaque point en 16 codes différents. Ce recodage était ensuite utilisé par des arbres binaires de décision pour réaliser la classification proprement dite. Cette méthode avait été initialement développée pour la reconnaissance de caractères [1], car elle permet de déterminer automatiquement quelles sont les configurations locales les plus intéressantes (courbes, points de rebroussement, etc.) dans des images binaires à très faible résolution, comme celles recontrées dans le domaine de la reconnaissance d'écriture.

Le problème de la détection de visages, et plus généralement de personnes, dans des séquences vidéo (cf. figure 1) offre un cadre un peu différent. D'une part les images sont de tailles beaucoup plus importantes (la résolution augmente d'un facteur 10, et est ici de 384×288), d'autre part la topologie des contours est beaucoup plus régulière, et enfin les pixels ne sont plus simplement binaires, mais portent des niveaux de gris (ici quantifiés en 256 valeurs).

Nous avons donc développé un algorithme de détection de contours original qui détecte les bords, et quantifie en 16 classes les directions des tangentes (cf. fig 2). Pour être utilisable sur des séquences vidéos, cet algorithme se devait d'être très robuste au bruit. Pour cela on utilise une stratégie multi-résolution :

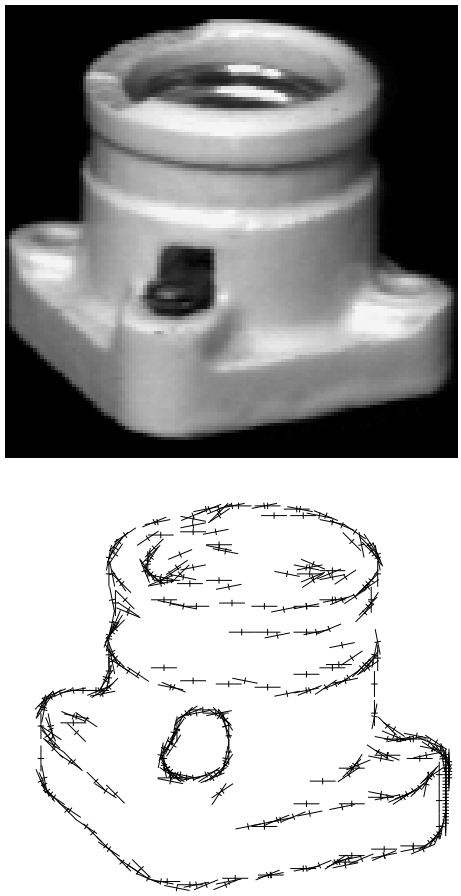

FIG. 2 – *La procédure de recodage détecte les bords et détermine leurs orientations*

En chaque point (x, y) de l'image, pour chaque direction $\alpha \in \{0, \frac{1}{16}\pi, \ldots, \frac{15}{16}\pi\}$ et pour chaque diamètre $d \in \{3, 7, 11\}$, on estime quelle est la proportion p de points du disque centré en (x, y) et de rayon d qui ont un niveau de gris supérieur à celui de leurs symétriques par rapport à la droite passant en (x, y) et de direction α (cf. fig 3).

Si cette proportion atteint un seuil P_{min} pour tous les diamètres, un bord est détecté, et la direction du bord est celle qui maximise cette proportion.

Cet algorithme ne fait intervenir que des comparaisons entre pixels, et est donc stable pour n'importe quelle transformation croissante des niveaux de gris. Ainsi, des variations dans l'éclairage ne modifient que faiblement l'image des contours.

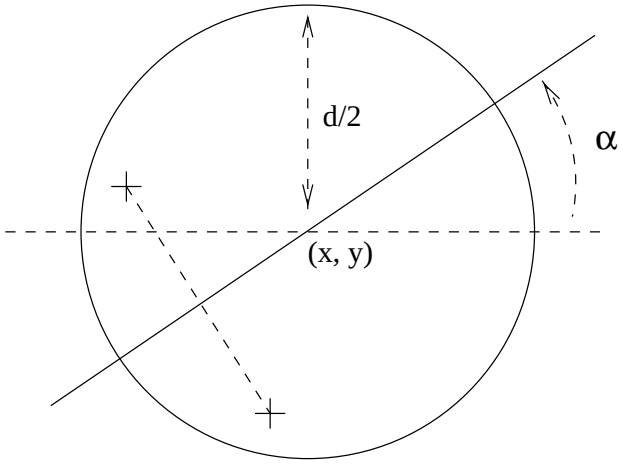

FIG. 3 – *La détection de contours multi-résolution consiste à estimer en chaque point, pour plusieurs diamètres et plusieurs directions, la proportion de pixels qui ont un niveau de gris très différent de leur symétrique.*

3 Gestion des contours illusoires

La détection du visage sur la séquence vidéo s'effectue après un pré-traitement qui élimine les parties statiques de la scène. Cette segmentation par champ de vitesse est réalisée à l'aide d'un algorithme développé par l'équipe TEMICS de l'IRISA (voir [8] pour une explication détaillée). Elle consiste à générer une partition de l'image en régions où le champ de mouvement suit un modèle paramétrique affine uniforme. La segmentation obtenue à l'image $t-1$ sert de support initial au calcul robuste du mouvement affine dans chaque région, puis cette segmentation est mise à jour grâce à l'analyse locale du champ calculé (modification des contours des régions existantes, apparition et estimation du mouvement sur les nouvelles régions). Dans notre cas, on retient la totalité des régions dont le mouvement est supérieur à un seuil empirique, fixe pour toute la séquence. Bien que fournissant une segmentation approximative, où le mouvement intrinsèque du personnage et le mouvement de la caméra peuvent être confondus, ce traitement permet de restreindre le travail de reconnaissance des arbres binaires aux parties de l'image potentiellement porteuses d'information. Il arrive néanmoins que le sujet devienne immobile pendant de courts instants, ce qui nécessite de compléter l'analyse dynamique par un post-traitement (voir section 6).

Pa ailleurs, la détection des bords a été modifiée pour ne pas générer des bords illusoires qui correspondent à la frontière entre la zone en mouvement et la zone statique. Pour que ces bords n'apparaissent pas sur l'image recodée, l'algorithme de détection ignore les points dont le voisinage mord sur la zone statique de l'image (celle qui a été éliminée par la segmentation par champs de vitesses), voir figure 5.

FIG. 4 – *La segmentation par champs de vitesses (image du haut) élimine les parties immobiles de l'image.*

4 Détection à l'aide d'arbres de classification

4.1 Reconnaissance

Un arbre binaire de classification peut être vu comme une représentation abstraite d'une stratégie du *jeu des vingts questions*. Dans ce jeu qui se joue à deux, l'un des joueurs doit deviner un mot choisi par l'autre joueur, en posant au maximum vingts questions dont les réponses peuvent être "oui" ou "non". Dans un tel jeu, la question posée à un moment donné dépend des questions et réponses antérieures.

Pour représenter une telle stratégie, chacun des noeuds internes d'un arbre binaire de décision porte une question et possède deux branches correspondants aux deux réponses possibles, et chaque feuille porte une étiquette indiquant la réponse finale. Ici il n'y a que deux étiquetages possibles : *visage présent* et *visage absent*. Chacune des questions placées aux noeuds internes teste la présence dans l'image d'un arrangement géométrique entre deux bords.

Dans le programme qui a été développé pour les expérimentations présentées dans cet article, les arrangements sont paramétrés par quatres angles γ_1, γ_2, α et β. L'arrangement défini par ces paramètres est présent dans l'image s'il existe deux pixels, l'un dont le voisinage porte un bord avec une orientation γ_1, l'autre un bord avec une orientation γ_2, et tels que le vecteur qu'ils définissent dans le plan image fasse un angle avec l'horizontale compris entre $\alpha - \beta$ et $\alpha + \beta$ (cf. figure 6).

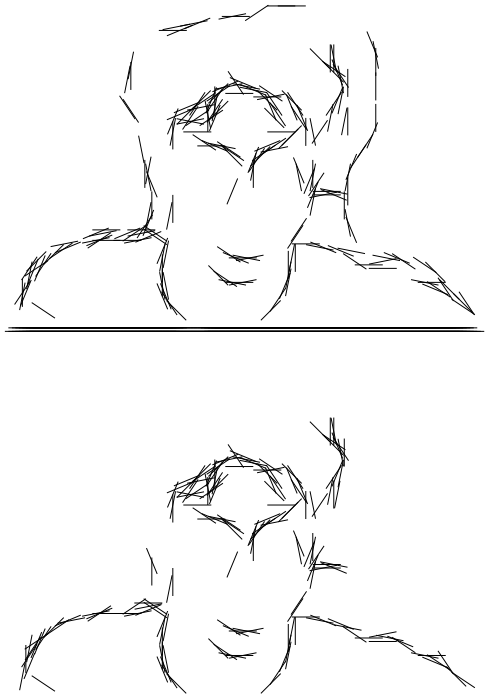

FIG. 5 – *La segmentation par champs de vitesses crée des bords illusoires (image du haut). Pour les éliminer, on ignore les bords au voisinage de la zone statique (image du bas).*

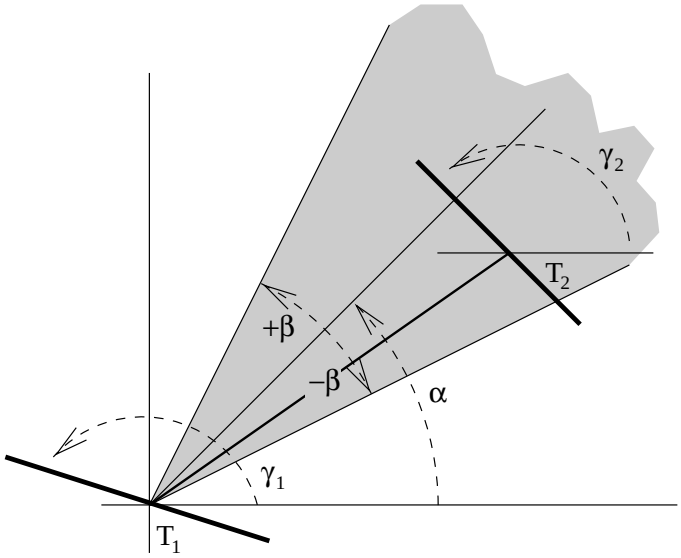

FIG. 6 – *Les relations recherchées à chaque nœud sont de la forme "existe-t-il un pixel portant le code T_1 et un pixel portant le code T_2 tels que le segment qu'ils définissent fasse un angle compris entre $\alpha - \beta$ et $\alpha + \beta$ avec l'horizontale".*

Hormis pour le premier nœud de l'arbre, on rajoute comme contrainte que l'un de deux pixels intervenant dans chaque arrangement est identique à un pixel utilisé dans un arrangement trouvé dans un noeud antérieur.

Dans le cas où l'arrangement associé à une des questions est présent dans l'image, nous pouvons lui associer une arête de graphe dont les deux sommets sont localisés sur les pixels intervenant dans la question. De plus, comme l'arrangement recherché à chaque nœud fait intervenir un pixel présent dans l'arrangement d'un nœud antérieur, ce graphe est connexe. Et parce que chaque arrangement ne fait intervenir que deux pixels, dont un nouveau, il est également sans cycle. Le nombre de sommets de ce graphe dépend du nombre de réponses positives dans les nœuds précédents et de la profondeur du nœud. Une image qui répond "oui" à chaque nœud porte un graphe qui a autant de sommets que de nœud visités plus un (puisque le premier nœud fait intervenir deux nouveaux pixels), alors qu'une image qui répond toujours "non" porte un graphe vide.

Les arrangements recherchés par les noeuds peuvent être trouvés en plusieurs endroits sur l'image. Il y a donc plusieurs "instances" pour chaque réponse positive (cf. fig 7). La principale différence avec les techniques standards d'arbres de décision réside dans cette "instanciation" des réponses. Nous verrons dans le paragraphe 5 que cette instanciation peut être utilisée pour compléter l'information de reconnaissance par une information en localisation.

4.2 Construction des arbres

La construction d'un arbre se fait de manière récursive à l'aide d'une base d'apprentissage. Cette base contient plusieurs dizaines d'images de visage, et plusieurs dizaine d'images diverses sur lesquelles ne se trouvent pas de visages. Elle est utilisée pour estimer la probabilité qu'une image qui a répondu "oui" à plusieurs questions soit une image contenant un visage. Cette estimation se fait simplement en comptant le nombre N_0 d'images sans visage de la base de données qui répondent "oui", le nombre N_1 d'images de visages qui répondent "oui" et en calculant le ratio :

$$\frac{N_1}{N_0 + N_1}$$

La procédure récursive choisit, étant données les images de la base d'apprentissage qui y arrivent, si un noeud doit être un noeud terminal (feuille) ou un noeud interne, et dans ce dernier cas quelle est la question à y placer. Ainsi, la construction s'amorce en utilisant

FIG. 7 – *A chaque feuille d'un arbre de classification correspond un graphe qui peut être présent en plusieurs endroits sur l'image. Les sommets de ce graphe sont positionnés sur des pixels. Les arêtes indiquent quels sont les couples de pixels liés par une contrainte géométrique. Figure du haut : Une instance du graphe sur une image de voiture. Figure du bas : Différentes instances du même graphe.*

l'intégralité de la base pour choisir le premier noeud, c'est-à-dire la racine de l'arbre. Ensuite, si nécessaire, on rappelle cette routine pour construire chacune des deux noeuds fils, en utilisant pour chacun le sous-ensemble de la base d'apprentissage qui répond l'une ou l'autre des réponses. On répète ce schéma jusqu'à ce que l'arbre soit complet.

La procédure proprement dite fonctionne de la manière suivante : si les images qui arrivent au noeud à créer sont toutes de la même classe ("visages" / "non visage"), alors ce noeud sera une feuille qui portera comme étiquette la classe des images qui y arrivent. Dans le cas contraire, le sommet est un nœud interne, et il faut déterminer la question qu'il doit porter, c'est-à-dire l'arrangement à rechercher sur les images qui arrivent là. Ce choix se fait en générant un ensemble aléatoire $\mathcal{A}$ d'arrangements candidats, ensemble qui constitue un échantillon de tous les arrangements possibles.

On peut estimer pour chacun de ces candidats la quantité d'information qu'il fourni en utilisant l'entropie de Shannon. L'entropie permet d'évaluer à quel point une grandeur est aléatoire. On sélectionne donc des questions telles que la classe de l'image Y, qui peut être modélisée comme une variable aléatoire, conditionnée par la réponse à la question, soit le moins aléatoire possible. Si elle devient déterministe, alors il n'est plus nécessaire de poser des questions, et c'est donc le choix optimal. Nous utilisons donc comme estimateur de la qualité d'un arrangement l'entropie de la classe de l'image conditionnellement aux réponses antérieures : plus cette entropie est faible, meilleure est la question.

Comme nous l'avons vu, en utilisant les images de la base d'apprentissage, nous pouvons estimer les probabilité $\hat{P}_1$ qu'une image qui répond "oui" à plusieurs questions $Q_1, \ldots, Q_n$ soit une image de visage, et la probabilité $\hat{P}_0$ qu'elle ne contienne pas de visage. Or l'entropie de Shannon en base 2 de la classe Y se met sous la forme :

$$\hat{H}(Y) = \log_2(\hat{P}_0)\hat{P}_0 + \log_2(\hat{P}_1)\hat{P}_1$$

4.3 Utilisation de plusieurs arbres

Lors de la construction d'un arbre, nous avons utilisé à chaque nœud un sous-ensemble $\mathcal{A}$ de questions générées de manière aléatoire. Si nous utilisons des sous-ensembles différents, nous obtiendrons un arbre différent. Pour obtenir une robustesse supérieure, nous construisons ainsi plusieurs dizaines d'arbres dont nous combinons les résultats. Pour classifier une image, nous déterminons pour chacun des arbres à quelle feuille tombe l'image, et quelle est la classe associée à cette

feuille. Ainsi chaque arbre vote pour une classe, la classe remportant le plus de votes sera la classe résultat.
Cette robustesse est en partie dûe au fait que des arbres construits avec des ensembles de questions différents utilisent des parties différentes du visage à détecter. Ainsi, dans le cas d'une occultation partielle ou d'un bruitage localisé, un arbre seul pourrait échouer car il n'utilise que cette partie du visage. Plusieurs arbres explorent de manière plus exhaustive le plan image.
Notons que dans le cas où l'on utilise plusieurs arbres que l'on combine, il est plus intéressant de créer des arbres statistiquement indépendants entre eux, même s'ils sont individuellement peu performants, plutôt que de créer des arbres individuellement très performants mais statistiquement corrélés. La taille de l'échantillon $\mathcal{A}$ est le paramètre qui permet de choisir la qualité de la minimisation de l'entropie, et donc la puissance de classification des arbres pris individuellement : plus on utilise un $\mathcal{A}$ grand, plus on minimise l'entropie, et on génère donc des arbres très puissants individuellement mais peu différents.
Enfin, si V_0 est le nombre de vote pour la classe "sans visage" et V_1 le nombre de vote pour la classe "visage" ($V_0 + V_1$ est donc le nombre total d'arbres), on peut estimer une "probabilité" que l'image contienne bien un visage en calculant le ratio

$$P_V = \frac{V_1}{V_0 + V_1}$$

Bien que cela n'ait pas été utilisé ici, il est très facile de rajouter un critère de rejet basé sur la cohérence des réponses des différents arbres. Il suffit pour cela de fixer un seuil que doit atteindre cette probabilité, ou celle de l'absence de visage :

$$\max(P_V, 1 - P_V) \geq \sigma$$

Dans le cas où ce seuil n'est pas atteint, cela signifie qu'aucune des deux hypothèses n'a obtenu suffisamment de votes, en fonction de la finalité de la détection, on pourrait faire appel à un opérateur humain ou à un autre algorithme.

5 Localisation

Comme nous l'avons vu, la détection consiste à trouver sur l'image des arrangements de bords significatifs, qui indiquent sans ambiguïté la présence de l'objet recherché (ici le visage de la personne). On peut à l'aide de la localisation de ces arrangements déterminer la position de l'objet visage.
Plus précisément, lors de la phase de détection, chaque arbre recale sur l'image un graphe dont les sommets sont placés sur des bords qui doivent être dans des directions données, et dont les arêtes vérifient des contraintes d'orientations. Pour une image donnée, on peut donc associer à chaque arbre $\mathcal{T}$ le plus petit rectangle

$$(x_{min}, y_{min}, x_{max}, y_{max})$$

contenant tous les sommets des graphes (cf. fig 8).
Ainsi, considérant l'ensemble des arbres $\{\mathcal{T}_1, \ldots, \mathcal{T}_n\}$, à chaque image correspond une famille de rectangles

$$\{(x^1_{min}, y^1_{min}, x^1_{max}, y^1_{max}), \ldots, (x^n_{min}, y^n_{min}, x^n_{max}, y^n_{max})\}$$

nous définissons la boîte englobante de l'objet détecté comme le rectangle

$$(\underline{x_{min}}, \underline{y_{min}}, \underline{x_{max}}, \underline{y_{max}})$$

où on note $\underline{x}$ la médiane des échantillons (cf. fig 9). L'utilisation de la médiane permet d'obtenir un résultat qui est stable même si certains sommets sont mal localisés.

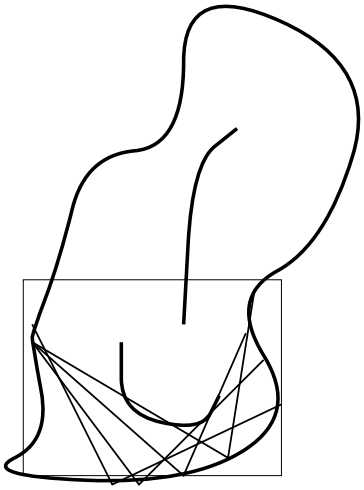

FIG. 8 – *Chaque arbre recale un graphe en plusieurs endroits sur l'image, on peut en déduire une boîte englobante. La figure du haut montre une des instances du graphe, celle du bas toutes les instances.*

6 Exploitation de la cohérence temporelle

La segmentation par champs de vitesses utilisée comme pré-traitement élimine les zones statiques, bien qu'elles soient parfois porteuses d'information. En particulier il peut arriver que la personne que l'on cherche dans l'image soit immobile pendant un court instant. Dans

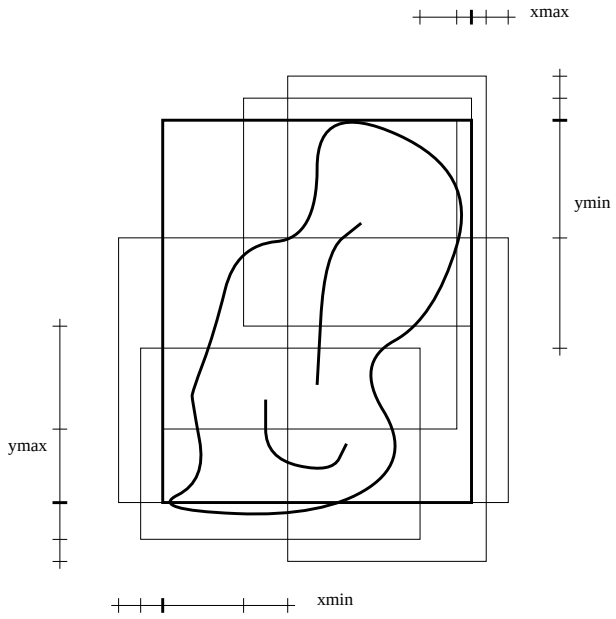

FIG. 9 – *Considérant toutes les boîtes, on en déduit une boîte globale dont les coordonnées des sommets sont les médianes des coordonnées des boîtes intiales.*

un tel cas, elle aura été éliminée de l'image segmentée, et ne peut évidemment être détectée lors de la première phase de l'analyse. Pour corriger ce défaut, on utilise le fait que par définition si la personne est masquée accidentellement, cela signifie qu'elle n'a pas bougé, et qu'elle se trouve à la même place que la dernière fois qu'elle a été détectée.

Précisément, pour chaque image pour laquelle la détection échoue, on réitère l'algorithme sur la sous-image délimitée par la boîte englobante calculée lors de la dernière détection. Cette sous-image est prise dans l'image complète, c'est-à-dire dans l'image telle qu'elle est avant la segmentation. Ce deuxième passage diminue dans des proportions importantes les erreurs de type faux négatifs (présence non détectée) en éliminant la plupart des erreurs provoquée par une segmentation abusive.

7 Résultats

La séquence étudiée est une séquence de 1035 images de qualité vidéo amateur, encodées en MPEG-1, fournies par le laboratoire TEMICS. Cette séquence est constituée de 3 sous-séquences: deux plans où un sujet vient s'asseoir à un bureau et parle à la caméra avec diverses mouvements (agitation du tronc, des bras, rotation de la tête, zoom de la caméra, etc.), séparés par une séquence où le sujet est absent et la caméra fait un panoramique sur le décor. Nous avons réduit la taille

FIG. 10 – *Utilisation de la cohérence temporelle. Haut-gauche : segmentation d'une image en milieu de séquence, l'algorithme détecte un visage malgré une segmentation déjà fortement dégradée. Haut-droit : Image suivante, le sujet s'immobilise, on ne détecte plus le visage. Bas-gauche : la boîte englobante détectée sur la dernière image reconnue. Bas-droit : l'imagette découpée grâce à cette boîte sur l'image statique permet de nouveau la reconnaissance du visage.*

des images à la résolution 384×288 et nous les avons converties en noir et blanc (cf. fig 11). L'apprentissage a consisté à créer 100 arbres binaires dont la profondeur est une douzaine de questions (la construction de nouvelles questions se poursuit tant que le nombre d'images disponibles pour les estimations statistiques est supérieur à trois). La base de données utilisée pour l'apprentissage contient 100 images sans la personne à détecter (ces images sont pour partie issue de la base Colombia, et pour partie des morceaux de fond), et 100 images de la personne (dans différentes positions - profil, face, trois-quart, etc. - et à différentes échelles, sur un fond neutre).

Le taux de reconnaissance de la classe "visage", sans le raffinement utilisant la cohérence temporelle réalise un score de 14% de faux négatif (visage présent non détecté. La plupart du temps à cause d'une trop grande immobilité du sujet), et 0.7% de faux positif (détection du visage alors qu'il n'est pas présent dans l'image). Avec le raffinement utilisant la cohérence temporelle, le taux de faux négatif chute à 2%, et le taux de faux positifs à 0.2%. La détection échoue donc sur 11 images où le sujet est présent, essentiellement à cause d'une trop grande immobilité de l'acteur couplée à une mauvaise estimation de la boîte englobante, qui met en échec le raffinement utilisant la cohérence temporelle. Le taux de faux positifs est très faible, puisqu'il cor-

respond à 5 images sur 1035. Ce résultat est dû en grande partie à la segmentation par le mouvement qui élimine une grande partie du fond parasite.

Si l'on examine plus finement les résultats en considérant les distributions a posteriori, c'est-à-dire les probabilités estimées que l'image contienne ou pas la personne, on note clairement que les probabilités sont nettement contrastées sur des images non ambiguës (acteur de face, scale correspondant aux images d'apprentissage), et elles deviennent plus similaires sur des images ambiguës (zoom en gros plan, visage incliné ou partiellement masqué). On peut donc imaginer facilement une nouvelle amélioration utilisant le critère de rejet (cf. section 4.3).

FIG. 11 – *Détection et localisation du visage dans la séquence vidéo* Armel.

8 Conclusion

Nous avons montré avec cette application l'intérêt de l'utilisation des arbres binaires de décision pour la détection et la localisation d'un visage dans une séquence vidéo. L'instanciations des réponses, et l'utilisation dans ce cas de la sémantique géométrique associée (chaque instance représente un positionnement spatial de la réponse dans le plan image) nous a permis d'obtenir en plus d'une information sur la classe de l'image, un positionnement de l'objet reconnu.

Enfin, l'exploitation de l'information temporelle, et de la cohérence de la segmentation par champs de vitesses a diminué fortement le nombre de visages non détectés.

La robustesse de cette technique statistique, alliée à l'utilisation des spécificités de la vidéo (possibilité d'utiliser une segmentation par champs de vitesses et la cohérence des déplacements) ont permis d'atteindre d'excellents taux de détection.

Annexe

Références

[1] Y. Amit, D. Geman, and K. Wilder. Recognizing shapes from simple queries about geometry. Technical report, Univ. of Chicago, 1995.

[2] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, CA., 1984.

[3] W. Buntine and T. Niblett. A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8:75–85, 1992.

[4] G. Burel and D. Carel. Detection and localization of faces on digital images. *Pattern Recognition Letters*, 15:963–967, 1994.

[5] L. A. Clark and D. Pergibon. Tree-based models. In J. M. Chambers and T. J. Hastie, editors, *Statistical Models in S.* Wadsworth & Brooks, Pacific Grove, CA, 1992.

[6] B. Jedynak and F. Fleuret. Reconnaissance d'objets 3d à l'aide d'arbres de classification. In *Proc. Image'Com 96*, Bordeaux, France, 1996.

[7] B. Moghaddam and A. Pentland. Probabilistic visual learning for object representation. *IEEE Trans. PAMI*, 19:696–710, 1997.

[8] J.M. Odobez and P. Bouthemy. Separation of moving regions from background in an image sequence acquired with a mobile camera. In H. Derin H.H. Li, S. Sun, editor, *Video Data Compression for Multimedia Computing*, pages 283–311. Kluwer Academic Publisher, 1997.

[9] E. Osuna, R. Freund, and F. Girosi. Training support vector machines: an application to face detection. In *Proceedings, CVPR*, pages 130–136. IEEE Computer Society Press, 1997.

[10] J.R. Quinlan. Induction of decision trees. *MachineLearning*, 1:81–106, 1986.

[11] J.R. Quinlan and R.L. Rivest. Inferring decision trees using minimum description length principle. *Information and Computation*, 80:227–248, 1989.

[12] B. D. Ripley. Neural networks and related methods for classification. *J. Royal Statist. Soc., B*, 56:409–437, 1994.

[13] H. A. Rowley, S. Baluja, and K. Takeo. Neural network-based face detection. *IEEE Trans. PAMI*, 20:23–38, 1998.

[14] K. K. Sung and T. Poggio. Example-based learning for view-based human face detection. Technical Report A.I Memo 1521, Artificial Intelligence Laboratory, M.I.T, 1994.

[15] K. K. Sung and T. Poggio. Example-based learning for view-based face detection. *IEEE Trans. PAMI*, 20:39–51, 1998.