Recherche d'extraits vidéos par reconstitution des trajectoires de caméras mobiles à partir d'un modèle spatio-temporel - Application à la vidéosurveillance.

Dana Codreanu [1], **André Péninou** [1], **Florence Sèdes** [1]

1 IRIT, Université Paul Sabatier, 118 Route de Narbonne
Toulouse, France
dana.codreanu,andre.peninou,florence.sedes@irit.fr

RÉSUMÉ. *Le domaine d'application de ces travaux relève de l'aide aux opérateurs humains de vidéosurveillance dans l'analyse manuelle d'extraits vidéos particuliers (e.g., recherche individu, objet perdu) via la sélection automatique d'un ensemble de caméras susceptibles d'avoir filmé une scène recherchée et l'identification des séquences vidéos correspondant à chaque caméra. Cette sélection s'appuie sur un modèle de données dont la contribution consiste à rendre disponibles des données de calcul de la géométrie du champ de vue des caméras. Plutôt que de stocker cette géométrie en tant que telle et son évolution au cours du temps, celle-ci est calculée dynamiquement en fonction de caractéristiques temporelles. Nous proposons un opérateur de sélection de caméras basé sur des critères spatiotemporels de calcul des intersections entre, d'une part, un parcours donné (e.g., personne agressée), et, d'autre part, les prises de vues des caméras fixes et les trajectoires des caméras mobiles. Nous illustrons l'opérateur par des exemples.*

ABSTRACT. *The scope of application of this work concerns the assistance to the operators of videosurveillance in the search for particular videos (e.g., attack of persons, lost object) by the way of automatic identification of a set of cameras likely to have filmed a required scene. This search is based on a multi-layer modeling whose characteristic consists in rather providing calculation data of the geometry of the cameras viewpoints on the network layer than to store this geometry as such. We propose an operator of selection of cameras based on spatiotemporal criteria in order to calculate the possible intersections between, on the one hand, a given journey (attackedperson), and, on the other hand, the shootings of fixed cameras and the trajectories of the mobile cameras. We illustrate the algorithms by a use case and examples of requests.*

MOTS-CLÉS : *vidéosurveillance, trajectoire des caméras, champ de vue*

KEYWORDS: *videosurveillance, cameras trajectory, field of view*

1. Introduction

Les capteurs vidéos sont très répandus de nos jours, générant des volumes de données impressionnants. Le nombre de caméras de vidéosurveillance déployées dans les grandes villes censées assurer la sécurité des citoyens augmente régulièrement. Ainsi, le réseau de vidéosurveillance de la RATP génère plusieurs péta-octets de données de capture vidéo par jour. Les vidéos sont stockées sur des serveurs et analysées depuis des centres de contrôle. Aujourd'hui, cette analyse est purement manuelle et elle est réalisée par des opérateurs qui scrutent plusieurs écrans disposés en matrice (mur d'images).

En conséquence, de nombreux travaux sont menés pour le développement de "systèmes de vidéosurveillance intelligents". Beaucoup de travaux visent le développement d'outils d'analyse du contenu (Cucchiara, 2005) mais les problèmes de volumétrie, d'hétérogénéité de contextes d'acquisition, de qualité des enregistrements et de manque d'accessibilité à certains contenus (e.g., vidéos privées) rendent inutile voire impossible l'exécution des algorithmes d'analyse du contenu à cause des mauvaises performances ou de la faible qualité des résultats obtenus. Le besoin persiste de proposer des approches de filtrage, de modélisation, d'indexation et de recherche dans des collections de contenus vidéos sans avoir recours à une indexation exhaustive basée contenu. L'idée principale est d'utiliser des métadonnées provenant d'autres sources (e.g., données issues des capteurs, caractéristiques techniques, annotations sociales) pour générer des descripteurs ou résumés.

Avec le développement des nouvelles technologies, il est devenu facile et peu onéreux de déployer d'autres types de capteurs (e.g., capteurs GPS, boussoles, accéléromètres) associés aux caméras. Les travaux existants montrent qu'en se basant sur les métadonnées spatiales et les caractéristiques de la caméra, il est possible d'extraire des informations précieuses et précises sur la scène filmée.

Dans cet article, nous proposons une méthode qui, à partir de segments de trajectoires (géolocalisation + timestamps [1]), sélectionne les caméras susceptibles d'avoir filmé une trajectoire donnée et identifie les séquences vidéo correspondant à chaque caméra selon l'intervalle de temps. Pour cela, nous proposons de nous appuyer sur un modèle de données contenant des informations spatiotemporelles pour reconstituer les trajectoires et les géométries des champs de vue des caméras de vidéosurveillance afin de permettre la sélection des caméras fixes et mobiles. Notre approche ne nécessite pas d'accéder aux contenus des extraits vidéo ni de procéder à une indexation quelconque. Elle permet de reconstituer automatiquement des trajectoires et des géométries à partir des données stockées dans le modèle pour la trajectoire et l'intervalle de temps considérés.

L'article est structuré de la façon suivante: dans la Section 2, nous présentons une vue d'ensemble du domaine de la vidéosurveillance et nous montrons comment l'ana-

1. estampille temporelle

lyse du problème d'intersection des champs de vues des caméras avec la trajectoire de l'objet cible nous a menés à le traduire dans un problème de modélisation et interrogation de bases de données spatiotemporelles. Dans la Section 3, nous présentons un état de l'art des travaux qui utilisent l'information spatiotemporelle dans la recherche des vidéos en expliquant pourquoi ceux-ci ne peuvent pas être utilisés dans la vidéosurveillance. En se basant sur la littérature, nous proposons un modèle de données (Section 4.1) et des spécifications d'opérateurs de sélection de caméras (Section 4.2). Nous montrons quelques exemples de requêtes dans la Section 5. Nous finissons par des conclusions et des perspectives (Section 7).

2. Contexte de la vidéosurveillance

En étudiant la façon dont une requête est analysée aujourd'hui dans les systèmes de vidéosurveillance, nous avons fait plusieurs observations. Les images restituées via le mur d'images ne sont pas organisées spatialement. Très peu d'informations ou métadonnées (un identifiant de la caméra, un timestamp, éventuellement une localisation) sont disponibles pour l'opérateur qui ne peut que se référer au numéro de la caméra et à son expertise personnelle pour situer celle-ci en fonction des éléments de la requête de la victime ou de l'enquêteur dans le but de reconstituer le cheminement de la victime qui constitue le point de départ de toute recherche. Plusieurs études évoquent les problèmes de surcharge cognitive, fatigue et ennui qui peuvent entraîner des erreurs et des temps de traitement allant jusqu'à plusieurs jours (Keval, 2009).

Les enregistrements issus des systèmes de vidéosurveillance n'ont pas de sens sans informations de localisation spatiotemporelle : le besoin le plus fréquent consiste à retrouver des vidéos liées à une certaine région ou segment de rue pendant un intervalle de temps. La recherche effectuée par les opérateurs concerne donc un lieu relatif à un réseau routier ou un réseau de transport par exemple et un intervalle de temps. Une telle requête représente le point d'entrée de toute enquête. Elle décrit aussi le cheminement de l'objet cible (e.g., personne), c'est à dire un ensemble de positions à des temps donnés. La recherche porte donc sur une requête définissant une trajectoire et un intervalle de temps.

Notre but est, à partir d'une cartographie de l'ensemble des caméras de vidéosurveillance (peu importe le système auquel elles appartiennent), de pouvoir sélectionner de façon automatique les caméras qui ont pu capturer des informations pertinentes sur une trajectoire donnée.

3. État de l'art sur l'annotation et la recherche de flux vidéo

L'état de l'art sur l'analyse des vidéos basée sur les métadonnées spatiotemporelles concerne différentes approches telles que l'annotation des images, le développement de systèmes d'aide à la décision basés sur l'interrogation des informations géospatiales ou des applications pour la gestion du trafic routier. Ces approches se différentient par :

– les métadonnées sur lesquelles ces systèmes se basent : position, géométrie de la scène observée, temps, caractéristiques techniques de la caméra ;

– la prise en compte du réseau routier ou de transport ;

– la représentation de ces données continues/discrètes, e.g. champ de vue de la caméra représentée comme une région mobile (une géométrie) qui est calculée pour chaque frame [2]/minute ou seconde ;

– le(s) type(s) de requête auxquels le système permet de répondre, e.g., Key Words, Region Based, Shortest Path, Nearest Neighbor, Visibility.

Le tableau 1 présente une comparaison de quelques approches existantes en fonction des différents critères énumérés.

Dans (X. Liu *et al.*, 2009), les auteurs proposent un système (SEVA) qui annote chaque frame d'une vidéo par la localisation, le timestamp et les objets présents dans la frame. Le système est composé de: (1) une caméra vidéo, (2) une boussole numérique, (3) un système de localisation, (4) une radio wifi associée à la caméra. Les auteurs partent de l'hypothèse que tous les objets susceptibles d'être capturés sur les vidéos sont dotés d'un système qui leur permet de transmettre leur localisation (qui sera captée par la radio wifi). A partir de la localisation des caméras et de la localisation des objets, les images (frames de la vidéo) sont annotées par les objets qu'elles sont susceptibles de contenir. Des opérations d'interpolation, extrapolation, synchronisation temporelle et filtrage basées sur le champ de vue de la caméra sont réalisées pour affiner les annotations. Les auteurs partent de l'hypothèse forte que tous les objets sont munis d'un capteur qui permet au système d'avoir sa localisation à chaque moment, ce qui n'existe que dans des environnements contrôlés. Ils construisent également la géométrie du champ de vue pour chaque seconde de vidéo.

Dans (Shen *et al.*, 2011), une approche similaire à SEVA est présentée, avec les différences suivantes : (1) les objets ne doivent pas transmettre leur position et (2) leur géométrie est prise en compte et non seulement le point de localisation. Pour chaque seconde de la vidéo, les auteurs calculent le champ de vue associé à la caméra et interrogent deux bases de données extérieures (OpenStreetMaps et GeoDec) afin d'extraire les objets (e.g., bâtiments, parcs) qui se trouvent dans la scène filmée. La liste des objets est affinée en éliminant les objets qui ne sont pas visibles (en calculant une visibilité horizontale et verticale). Pour chaque objet une liste de tags est calculée à partir des ressources extérieures (e.g., localisation, mots clefs, tags extraits de la page wikipedia associée). Un classement des tags est effectué en se basant sur des critères spatiaux (la distance entre la caméra et le centre de l'objet, l'aire de l'objet dans le frame annoté, etc.). Ce système permet ensuite de retrouver des segments de vidéo à partir des requêtes textuelles en calculant une similarité entre les mots de la requête et les tags associés à chaque segment de vidéo de la base.

2. trame video

Dans (Shahabi *et al.*, 2010), les auteurs présentent un framework de visualisation des informations géospatiales liées aux caméras, images, messages (en fonction des sources de données et des contextes) pour l'aide à la décision. Leur principale contribution est d'avoir défini une architecture trois tiers qui, en se basant sur une base de donnée qui intègre des informations provenant de plusieurs sources (images satellitaires, cartes, GIS datasets, données temporelles, flux vidéos) répond à des requêtes spatiotemporelles. Un grand effort a été fait pour le développement d'une interface qui facilite une bonne visualisation et interaction (leur proposition intègre les solutions de visualisation existantes (Google Maps [3], Google Earth [4])) en améliorant l'interaction par le rajout d'une barre temporelle. Toutefois, le modèle de données et la façon dont les sources de données sont intégrées (e.g., vidéo streams) ne sont pas détaillés. Le système ne prend pas non plus en compte la géométrie des champs de vue des caméras, mais seulement les positions.

Dans (Debnath, Borcea, 2013), une approche d'annotation des images en se basant sur la localisation et l'orientation de la caméra est présentée. L'originalité est représentée par le fait qu'en plus de la localisation et des caractéristiques optiques de la caméra (angle de vue), le système proposé (TagPix) calcule une distance entre l'utilisateur et différents objets situés dans l'aire de visibilité de la caméra afin de choisir le tag le plus pertinent. Les principaux points de similarité avec notre approche sont le fait de calculer le champ de vue et la distance vues par la caméra sans avoir accès au contenu. TagPix vise l'annotation des photos donc ne considère pas la mobilité des objets et des caméras et des requêtes de type trajectoire.

Tableau 1. Comparaison des systèmes de recherche des contenus vidéo en se basant sur des métadonnées spatiotemporelles. Les abréviations de la colonne Requête du tableau correspondent aux types de requêtes présentés en début de section (KW: Key words, R: Region Based, SP: Shortest Path, NN:Nearest Neighbor, V:Visibility). Réseau R/T est une abréviation de Routier/Transport

Approche	Application	Réseau R/T	Type de Requête	Représentation de données
(X. Liu *et al.*, 2009)	Annotation	Non	KW	Geom(t, p, α, d, R)
(Shen *et al.*, 2011)	Annotation	Non	KW	Geom(t, p, α, d, R)
(Shahabi *et al.*, 2010)	Aide à la décision	O/N	R, SP, NN, V	Non
(Debnath, Borcea, 2013)	Tagging des paysages	Non	KW	Non

Le problème du développement des systèmes de videosurveillance intelligents a donné lieu à des nombreux projets de recherche comme VANAHEIM [5] qui propose

3. https://www.google.fr/maps/preview
4. http://www.google.com/earth/
5. http://www.vanaheim-project.eu/

une solution de sélection des images à montrer aux opérateurs humains en temps réel mais en se basant sur des algorithmes d'analyse du contenu vidéo. Les observations que nous pouvons faire sont que les systèmes présentés ne modélisent pas les éléments essentiels à prendre en compte dans le contexte de la vidéosurveillance : réseau routier, réseau de transport, caméras. Par ailleurs, les systèmes qui prennent en compte les géométries des scènes visualisées par les caméras construisent ces géométries pour chaque frame, ce qui est inenvisageable pour la vidéosurveillance à cause du coût de traitement et du fait que souvent le contenu vidéo n'est pas directement accessible (problèmes de droits d'accès).

En conséquence, nous allons proposer dans la suite de cet article une solution de sélection automatique des caméras fixes et mobiles dont le champ de vue a pu filmer une trajectoire donnée comme requête. Ce processus de sélection s'appuie sur un modèle de données qui permet la reconstitution des trajectoires et des géométries des champs de vue des caméras.

4. Proposition de modèle de données

Afin de réaliser notre objectif, nous proposons un modèle qui représente, d'une part les réseaux routier et de transport et d'autre part : (1) les caméras fixes et leur position sur le réseau routier, (2) les changements des caméras fixes dans le temps et (3) les caméras mobiles et leur attachement à un objet mobile donnant ainsi leur position dans le temps.

La modélisation des champs de vue des caméras par rapport à une carte nous permettra par la suite de : (a) modéliser les trajectoires des caméras mobiles, (b) modéliser les champs de vue et les distances maximales de détection des caméras fixes et mobiles et (c) sélectionner les caméras pertinentes pour une certaine trajectoire.

4.1. Représentation de données

Dans ce qui suit nous allons définir la représentation des données pour chaque composante de notre modèle.

4.1.1. Réseau Routier

Définition 1 : On définit un réseau routier G_R = (E,V) comme étant un graphe non orienté où E = $\{e_i$ / e_i=(v_j, v_k)$\}$ est un ensemble de segments de rue et V= $\{v_i\}$ est l'ensemble de croisements de ces segments (K. Liu *et al.*, 2012).

Cette modélisation nous permet de garder plusieurs niveaux de granularité du réseau routier (Sandu *et al.*, 2011). De cette façon, on peut considérer comme arête du graphe chaque segment de rue, les portions de rue entre les grandes intersections ou les rues entières. La dernière option est la plus proche de la façon dont les adresses sont exprimées dans la vie réelle par un numéro relatif à la rue entière (et non pas un segment) (e.g., Rue Alsace Lorraine no 15, Toulouse, France).

En conséquence nous allons définir deux types de positions: *une position géométrique* qui est une position 2D par rapport au système géodésique (des coordonnées GPS <lat, long>) et *une position symbolique relative au réseau routier* qui est une position similaire à l'adresse postale. Il existe des fonctions et des API publiques qui permettent de passer d'un type de position à un autre (les dénominations utilisées par Google sont Geocode et Reverse Geocode).

Les données qui composent cette couche peuvent ou non être stockées dans la même base de données que les autres. Il existe des travaux qui transforment tout le réseau routier sous forme de graphe et le stockent (e.g., (Brinkhoff, 2002)) et des travaux qui utilisent des bases de données extérieures comme OpenstreetMaps [6] (Shen *et al.*, 2011).

Définition 2 : Soit la fonction de mapping map0(position): positionsGPS $\rightarrow G_R$ qui donne la position sur le graphe G_R (e.g., Rue Montesquieu no 14) à partir d'une position GPS. (e.g., il existe plusieurs systèmes offrant ces fonctions que nous allons utiliser: geocoder [7], Google Maps [8]). Il existe bien sûr la fonction inverse $map0^{-1}$(adresse):G_R $\rightarrow$ positionsGPS (Reverse Geocoding de Google Maps).

4.1.2. Réseau de Transport

Dans le contexte de la vidéosurveillance, les requêtes sont très souvent liées au réseau de transport (e.g., sac oublié dans le bus) et il y a beaucoup de caméras mobiles installées à l'intérieur ou à l'extérieur des bus.

Définition 3 : On définit un réseau de transport G_T = (E_T ,V_T) comme étant un graphe non orienté où E_T =\{ e_{ti} / e_{ti}=(v_{tj} , v_{tk})\} est un ensemble de tronçons de réseau de transport et V_T= \{v_{ti}\} est l'ensemble des stations de bus.

Définition 4 : Soit la fonction de mapping map1(v_{ti}) : $V_T \rightarrow G_R$ qui donne la position sur le graphe G_R des noeuds du graphe G_T .

4.1.3. Objets

Les données concernant les *Objets* comprennent les positions géométriques ou symboliques des objets fixes et mobiles. Les *Objets Fixes* sont situés sur des segments de rue. Dans le cas des *Objets Mobiles*, leur position change dans le temps. Chaque objet transmet périodiquement sa position en fonction de différentes stratégies (e.g., chaque n secondes, chaque fois que l'objet change de segment) que nous ne traitons pas dans cet article. Nous supposons que les remontées sont assez fréquentes et que nous avons au moins une position par segment de rue. Dans notre modèle nous distinguons deux types d'objets mobiles : (1) objets qui se déplacent librement dans le réseau routier (e.g., voiture, personne) et (2) objets dont les trajectoires sont contraintes par une ligne (e.g., bus).

---

6. http://www.openstreetmap.org/
7. http://geocoder.us/
8. https://www.google.fr/maps/

Définition 5 : Soit MO={mo} l'ensemble d'objets mobiles. Soit id(mo)=mo_i l'identifiant de l'objet mobile. Soit la fonction TR(mo_i) = {(position(mo_i), time(mo_i))} qui extrait la trajectoire de l'objet mobile mo_i. Soit $\{position_j(mo_i)\}$ l'ensemble de positions de l'objet mobile mo_i. Soit $\{time_j(mo_i)\}$ l'ensemble des timestamps de l'objet mobile mo_i.

4.1.4. Caméras

Au-dessus de toutes ces données, nous modélisons les *Caméras*. Cette couche est composée des caméras fixes et mobiles. Les caméras fixes ont une position 2D fixée au moment de l'installation. Les caméras mobiles sont associées à un objet mobile (e.g., bus) et leur trajectoire est la même que celle de l'objet.

Les caméras de nouvelle génération possèdent des capteurs GPS incorporés et même des boussoles. Les technologies développées autour de ces caméras rendent possible l'extraction automatique des caractéristiques de prise de vue, par exemple : l'orientation, le zoom, la distance focale, etc. En se basant sur ces éléments, il est possible de modéliser le champ de vue et de tracer ces modifications dans le temps. Le champ de vue est calculé à partir de cinq paramètres principaux (Ay *et al.*, 2010) : la position, l'angle de vue, l'orientation, la distance visible et la taille du capteur.

Définition 6 : Soit l'ensemble de caméras fixes FC = {fc} / fc est une caméra fixe, id(fc)=c_i donne son identifiant, position(c_i) donne sa position et FOV(c_i) donne l'ensemble de changements de son champ de vue.

Définition 7 : Soit l'ensemble de caméras mobiles MC = {mc} / mc est une caméra mobile, id(mc)=c_i donne son identifiant, mo(c_i)=moi$\in$MO donne l'objet mobile auquel la caméra est associée. La trajectoire de la caméra mobile c_i sera celle de l'objet mobile avec l'identifiant mo(c_i) donc TR(c_i)= TR(mo(c_i))).

4.2. Opérateur proposé

Dans le cadre de l'aide aux opérateurs de vidéosurveillance, nous supposons la requête traduite en :

– une trajectoire spatiale constituée d'une séquence de segments tr=(u_1, u_2, ... , u_n) projetés sur le réseau routier ;

– un intervalle de temps [t_1, t_2].

Les données peuvent être extraites à partir des données d'enquête et d'outils de manipulation de données géographiques (cf. section 4.1.1). Le but est de proposer à l'agent de vidéosurveillance une liste de séquences vidéo susceptibles de contenir la trajectoire recherchée. Pour cela, il nous faut rechercher les caméras dont le champ de vue intersecte la trajectoire dans l'intervalle de temps et donc susceptibles d'avoir filmé la scène recherchée.

En conséquence, nous proposons un nouvel opérateur appelé hasSeen. L'implémentation de l'opérateur pour les caméras fixes et mobiles est différente. Nous allons

présenter par la suite séparément les deux cas de sélection des caméras fixes et des caméras mobiles.

L'opérateur hasSeen se définit ainsi. Étant donné la trajectoire spatiale composée de segments tr=(u_1,..., u_n) et l'intervalle de temps [t_1, t_2], hasSeen(tr, t_1, t_2) retourne l'ensemble des caméras c_i (1<=i<=m) associées à un segment u_k et un extrait vidéo entre deux instants t^i_{start} et t^i_{end} avec t^i_{start}∈[t_1,t_2] et t^i_{end}∈[t_1,t_2]. Chaque element de cet ensemble indique que l'extrait vidéo entre t^i_{start} et t^i_{end} de la camera c_i a filmé le segment u_k. C'est donc l'ensemble des segments vidéos des caméras susceptibles d'avoir filmé la scène recherchée.

$$hasSeen : u_1, u_2, ..., u_n, [t_1, t_2] => \begin{cases} c_1 : t^1_{start} -> t^1_{end}, u_k (1 <= k <= n) \\ c_2 : t^2_{start} -> t^2_{end}, u_k (1 <= k <= n) \\ ... \\ c_m : t^m_{start} -> t^m_{end}, u_k (1 <= k <= n) \end{cases}$$

4.2.1. Caméras Fixes

Les résultats pour les cameras fixes seront l'ensemble de triplets: R = {r = (c_i, u_k, [t_a, t_b])}, c_i ∈ FixedCameras, u_k ∈ tr, $t_1 \leq t_a < t_b \leq t_2$. L'opérateur que nous avons défini vérifie quelles sont les caméras fixes dont le champ de vue a intersecté un des segments du trajet de la requête et entre quels moments de temps (les instants t_a et t_b).

r ∈ R ≡ Il existe $fov_j \in$ fov(c_i)/ (fov(c_i) est l'ensemble des instants de changement du champ de vue (Field Of View fov) de la caméra c_i) tel que :

time(fov_j) ∈ [t_1, t_2] ∧ intersects (u_k, geometry(position(c_i), fov_j)) ∧ t_a = time(fov_j) ∧ t_b = min (time (succ(fov_j)), t_2)	*dans le cas où le point de changement est à l'intérieur de l'intervalle [t_1, t_2], si la géométrie du champ de vue correspondant intersecte l'un des segments du trajet de la requête, alors l'intervalle de temps commence au moment du changement du champ de vue et se finit au prochain changement ou à la fin de l'intervalle de la requête*
∨	
time(fov_j)<t_1 ∧ t_1<=time(succ(fov_j)) ∧ intersects (u_k, geometry(position(c_i), fov_i)) ∧ t_a = t_1 ∧ t_b = min (time (succ(fov_i)), t_2)	*dans le cas où le point de changement est avant t_1, et son suivant est après t_1, si la géométrie du champ de vue correspondant intersecte l'un des segments du trajet de la requête, alors l'intervalle de temps commence au moment t_1 et se finit au prochain changement ou à la fin de l'intervalle de la requête*

4.2.2. Caméras Mobiles

L'opérateur que nous avons défini retrouve quelles sont les caméras associées à des objets mobiles dont une position connue intersecte un des segments du trajet de la requête et entre quels moments de temps il peut l'avoir intersecté (instants t_a et t_b). Dans ce cas, la mobilité des caméras ne permettent pas de calculer l'intersection précise entre le champ de vue de la caméra et les segments de la requête (par exemple une caméra placée sur le côté extérieur droit d'un bus). Le résultat obtenu indique uniquement que la caméra mobile se trouvait sur un segment de la requête dans l'intervalle de temps de la requête.

Les résultats pour les caméras mobiles seront l'ensemble de triplets: R = {r = (c_i, u_k, [t_a, t_b])}, $c_i \in$ MC, $u_k \in$ tr, $t_1 \leq t_a < t_b \leq t_2$.

r $\in$ R $\equiv$ Il existe $mp_j \in$ TR(mo(c_i)) (il existe une position dans la trajectoire de l'objet mobile auquel la caméra mobile c_i est attachée) tel que :

[time($mp_j(mo_i)$) $\in$ [t_1,t_2]	
$\wedge$ intersects (position($mp_j(mo_i)$), u_k)	
$\wedge$ (not intersects(	
prec(position($mp_j(mo_i)$)), tr)	
$\vee$ (intersects(	
prec(position($mp_j(mo_i)$)), tr)	
$\wedge$ prec(time($mp_j(mo_i)$))<t_1))	
$\wedge$ t_a=max (prec(time($mp_j(mo_i)$)), t_1)	*dans le cas où la remontée de la position de l'objet est sur le trajet de la requête et est à l'intérieur de l'intervalle [t_1, t_2] et la position d'avant n'est pas sur le trajet de la requête, alors l'intervalle de temps commence au maximum entre le moment de la dernière remontée et t_1 et se finit à la prochaine remontée ou à la fin de l'intervalle de la requête*
$\wedge$ t_b=min(succ(time($mp_j(mo_i)$)),t_2)]	
$\vee$ **// deuxième partie de l'opérateur**	
[t_1<= time($mp_j(mo_i)$)	
$\wedge$ time($mp_j(mo_i)$)<= t_2	
$\wedge$ intersects (position($mp_j(mo_i)$), u_k)	
$\wedge$ intersects(	
(prec(position($mp_j(mo_i)$))), tr)	
$\wedge$ t_a =time($mp_j(mo_i)$)	
$\wedge$ t_b=min(succ(time($mp_j(mo_i)$)), t_2)]	*dans le cas où la remontée de la position de l'objet est sur le trajet de la requête et est à l'intérieur de l'intervalle [t_1, t_2] et la position d'avant est aussi sur le trajet de la requête, alors l'intervalle de temps commence au moment de la remontée et se finit à la prochaine remontée ou à la fin de l'intervalle de la requête*

Nous définissons les fonctions et prédicats :

– *geometry(point p, fov f):* region calcule la géométrie du champ de vue d'une caméra à partir de sa position et de ses caractéristiques optiques (Ay *et al.*, 2010).

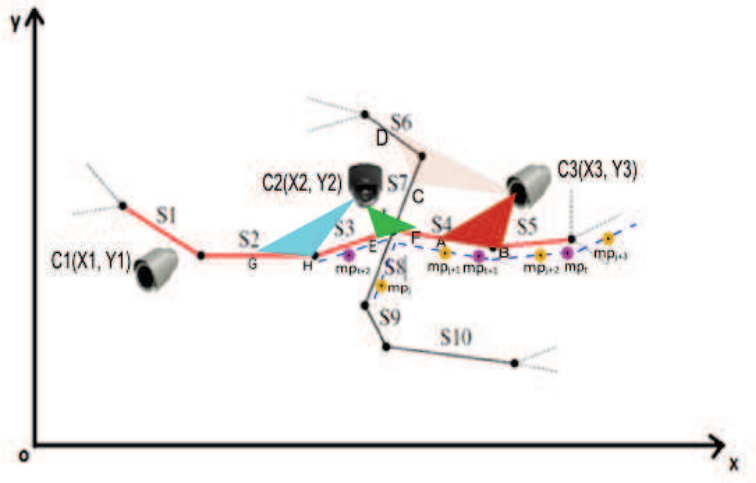

Figure 1. Schéma d'un réseau routier filmé par trois caméras fixes

Plutôt que de stocker cette géométrie en tant que telle et de la générer pour chaque frame de la vidéo comme le font les systèmes présentés dans l'état de l'art, elle est calculée en dynamique en fonction de caractéristiques temporelles au moment de la requête.

– *intersects(*line *seg,* region *g): boolean* permet de vérifier si la géométrie g intersecte un segment de rue.

– *intersects(*point *p,* line *seg): boolean* permet de vérifier si un point p intersecte un segment de rue.

– *intersects(*point *p, set(*line*) tr): boolean* permet de vérifier si un point p intersecte un ensemble de segments ; elle retourne vrai si $\exists\ seg_i \in$ tr / intersects (p, seg_i).

5. Exemples de requêtes

5.1. Caméras fixes

Supposons le schéma de la Figure 1 illustrant les localisations des caméras fixes C1, C2 et C3. Le réseau routier est représenté par les deux rues (S1-S5 et S6-S10). Supposons le trajet de la requête TR= S1,..,S5 et l'intervalle de la requête I= [t_1, t_2].

La Figure 2 présente les prises de vue des caméras C_2 et C_3 en fonction du temps. Les différents moments où les champs de vue des caméras C_2 et C_3 changent (voir Figure 1) sont marqués en couleurs correspondant aux géométries de la Figure 1 (e.g., au moment $time_j(fov(c_3))$ le champ de vue devient ABC_3).

Supposons l'intervalle de la requête [t_1, t_2] avec $time_j(fov(C_3))$<t_1<$time_k$ (fov(C_2)) et $time_{j+1}(fov(C_3))$< t_2.

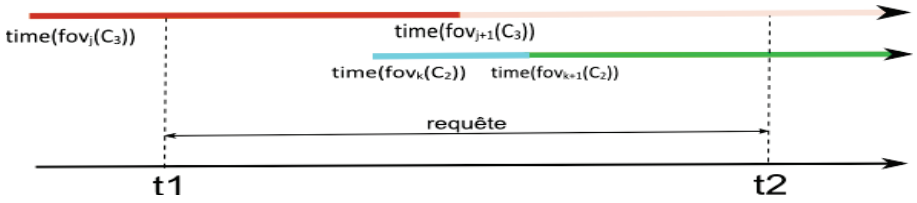

Figure 2. Les moments de changement de champ de vue et l'intervalle de la requête

La sélection des caméras fixes est réalisée par l'algorithme 1. Nous allons expliquer cet algorithme par l'intermédiaire de l'exemple illustré dans la Figure 1. Les premières lignes de l'algorithme (1-3) représentent une étape de filtrage. De toutes les caméras de la base de donnée nous allons sélectionner celles qui se situent à une distance maximale inférieure ou égale à la distance visible maximale des caméras de la base. Dans notre cas les seules caméras qui ont possiblement filmé les segments de la requête sont C_1, C_2 et C_3.

Pour chacune des caméras sélectionnées lors de la première étape, nous allons chercher les moments où celles-ci ont modifié leur champ de vue (field of view fov) (lignes 4,5 de l'algorithme). Les lignes 6-19 traitent les deux cas possibles: le changement est entre t_1 et t_2 (e.g., time($fov_k(C_2)$) ou le changement est avant t_1 (e.g., time($fov_j(C_3)$)). Les géométries des champs de vues sont construites et leurs intersections avec les segments de la requête sont evaluées.

Le résultat de la requête pour notre exemple est:
{(C_2, S_2, [time($fov_k(C_2)$), time($fov_{k+1}(C_2)$)]), (C_2, S_3, [time($fov_{k+1}(C_2)$), t_2])), (C_2, S_4, [time($fov_{k+1}(C_2)$), t_2))), (C_3, S_4, [t_1, time($fov_{j+1}(C_3)$)])}.

5.2. Caméras mobiles

Considérons le même schéma de la figure 1. Le but est maintenant de sélectionner les caméras mobiles susceptibles d'avoir filmé la trajectoire de la requête TR=S1, S2, S3, S4, S5. Cette sélection est faite selon l'algorithme 2.

Supposons deux objets mobiles ayant les trajectoires marquées en pointillé sur la figure (S8, S4, S5, ...). Nous supposons que l'objet envoie au moins une remontée mp_j (mobile position) contenant sa position et un timestamp par segment de rue. En considérant chaque segment de la rue et chaque objet mobile (lignes 1-2 de l'algorithme), la fonction $filtrer(mo_i, u_k, [t_1, t_2])$ va tester les cas expliqués dans 4.2.2 : la position de l'objet est sur la trajectoire de la requête et entre t_1 et t_2 (mp_t, mp_t, mp_{j+1}, mp_{j+2} comme illustré dans la Figure 3) et la position d'avant intersecte aussi (mp_{j+1} et mp_{j+2}) ou la position d'avant n'intersecte pas la trajectoire (mp_j et mp_{J+1}) ou elle intersecte mais avant t_1 (mp_t et mp_{t+1}).

Le résultat est {(obj_i, S_4, [t_1,time(mp_{j+1})]), (obj_i, S_5, [time(mp_{j+1}), t_2)], (obj_{i+1}, S_4, [time(mp_t),t_2)])}

Algorithm 1: L'algorithme de sélection des caméras fixes qui ont intersecté la trajectoire de la requête

Entrées: Une suite de segments de rue: u_k et un intervalle de temps: $[t_1, t_2]$.
Sorties: La liste des caméras fixes qui ont vu la trajectoire donnée

pour *chaque* u_k *de la requête* **faire**
  $listeCam \leftarrow extraireCamDist(u_k, max(FOV.visibleDistance))$
fin
pour *chaque* c_i *de listeCam* **faire**
  pour *chaque* $(fov_j(c_i))$ **faire**
    si *time*$(fov_j(c_i)) >= t_1$ *et time*$(fov_j(c_i)) <= t_2$ **alors**
      $geometry_{ij} \leftarrow construire\_polygone(fov_j(c_i))$;
      pour *chaque* u_k *de la requête* **faire**
        si $geometry_{ij}$ *intersecte* u_k **alors**
          $ajouter(c_i, u_k, [time(fov_j), min(succ(time(fov_j)), t_2)])$;
        fin
      fin
    fin
    si *time*$(fov_j(c_i)) < t_1$ *et* $t_1 <=$ *time*$(succ(fov_j(c_i)))$ **alors**
      $geometry\_ij \leftarrow construire\_polygone(fov_j(c_i))$;
      pour *chaque* u_k *de la requête* **faire**
        si $geometry_{ij}$ *intersecte* u_k **alors**
          $ajouter(c_i, u_k, [t_1, min(time(succ(fov_j)), t_2)])$;
        fin
      fin
    fin
  fin
fin

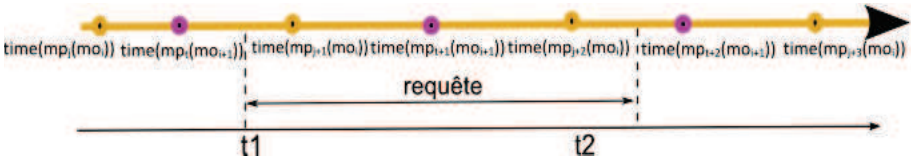

Figure 3. Les moments des trajectoires des objets mobiles et l'intervalle de la requête

6. Implémentation et validation

La figure 4 illustre l'architecture générique que nous avons implémentée et qui vise à faciliter la conception des outils forensic [9]. Nous allons brièvement présenter le

9. http://en.wikipedia.org/wiki/Forensic_science

Algorithm 2: L'algorithme de sélection des caméras mobiles qui ont intersecté la trajectoire de la requête

Entrées: Une suite de segments de rue: $u_k$et un intervalle de temps: $[t_1, t_2]$.
Sorties: La liste des caméras mobiles qui ont vu la trajectoire donnée

```
pour chaque u_k faire
    pour chaque mo_i faire
        listeObjMobiles ← ajouter(filtrer(mo_i, u_k, [t_1, t_2]));
    fin
fin
pour chaque mo_i.id de listeObjMobiles faire
    listeCameras ← selectionnerCameras(mo_i.id);
fin
```

choix d'implémentation que nous avons fait (dans la plupart des cas imposé par les contraintes du projet) mais qui n'est pas certainement le seul possible :

– **Terminal Interface (TI)** : Permet de saisir des requêtes à partir d'une interface basée sur l'API Google Maps.

– **Query Interpreter (QI)** : Transforme la requête de l'utilisateur dans une requête spatiotemporelle comme celle décrite dans la section 4.2 (une séquence de segments et un intervalle de temps).

– **SQL Query Generator (SQLG)** : Ce module implémente les deux algorithmes décrits dans la section antérieure. Il génère et envoie à la base de données les requêtes nécessaires pour le calcul des géométries des champs de vues des caméras fixes et leur intersection avec les segments de la requête.

– **Spatiotemporal database(DB)** : Ce module est responsable du stockage des données à partir desquelles le module SQLG reconstitue les trajectoire des objets mobiles et les géométries des champs de vue des caméras. Nous avons utilisé Oracle 11g (avec l'extension Spatial) pour implémenter ce module.

Les modules QI et SQLG sont implémentés en Java et le TI est implémenté en HTML/Javascript. Nous avons utilisé JSP et Ajax pour connecter l'interface avec les deux autres modules. Les tests sur des données expérimentales et données réelles sont en cours de mise en oeuvre.

7. Conclusion et perspectives

Dans cet article nous avons proposé une approche de sélection automatique des caméras susceptibles d'avoir filmé une trajectoire. Nous avons pu valider la contribution applicative de notre travail qui est de montrer la faisabilité et les bénéfices de l'utilisation des métadonnées (e.g., de géolocalisation, caractéristiques optiques de la caméra, réseau routier, réseau de transport) associées aux dispositifs de prise de vue dans un modèle qui constitue la base d'un système d'assistance aux opérateurs de vi-

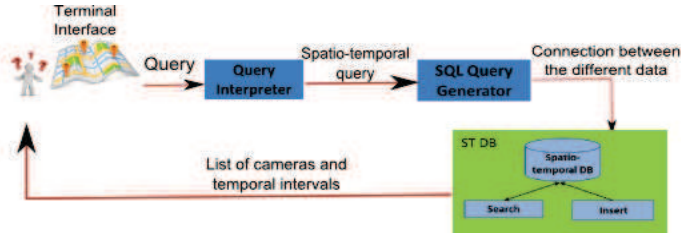

Figure 4. Architecture de l'outil proposé

déo surveillance. Cette approche permet de s'affranchir de l'indexation et de l'accès aux contenus vidéo, dans une première phase de recherche qui opère un filtrage important. Le besoin opérationnel exprimé dans cet article a été extrait de l'analyse du besoin et des entretiens avec les opérationnels (e.g., RATP, SNCF) réalisés dans le cadre du projet ANR METHODEO [10]. La solution proposée a été également implémentée et validée dans le cadre du projet dans un contexte industriel.

Les opérateurs de sélection proposés se basent sur un modèle de données spatiales et temporelles des caméras de vidéosurveillance dans le cadre d'un réseau de transport en commun urbain qui permet la reconstitution des trajectoires des objets mobiles des caméras mobiles (associées à des objets mobiles) et des positions des caméras fixes ; chaque caméra se voit associé un champ de vue dont la géométrie est calculée au moment de la requête pour les caméras fixes susceptibles d'avoir capturé des vidéos pertinentes pour la trajectoire cible. L'approche proposée, renforcée par des solutions d'indexation et de stockage optimisées et complétée par des solutions d'analyse du contenu vidéo, peut constituer la base d'outils de Forensic (Sèdes *et al.*, 2012) si recherchés dans le cadre de la vidéosurveillance.

Le résultat de l'opérateur proposé est une liste de segments vidéo dont les intervalles de temps peuvent se recouvrir. Une perspective concerne donc l'ordonnancement des résultats en fonction de la distance des caméras par rapport aux segments de la requête par exemple.

Pour l'instant notre modèle considère seulement les caméras situées en environnement outdoor qui ont des positions GPS qui permettent de faire la projection sur le graphe du réseau routier. Des travaux comme ceux de (Jensen *et al.*, 2009) modélisent également les environnements indoor sous forme de graphe (les noeuds du graphe sont représentés par des pièces et les segments par des connexions entre les pièces e.g., portes). Une perspective intéressante de notre travail est d'étendre le modèle pour prendre en compte les réseau de caméras à l'intérieur de stations de métro

10. http://www.agence-nationale-recherche.fr/suivi-bilan/ingenierie-procedes-securite/concepts-systemes-et-outils-pour-la-securite-globale/detail-des-projets-finances-csosg/?tx_lwmsuivibilan_pi2[CODE]=ANR-10-SECU-0006

ou des gares par exemple, le principal verrou étant de définir les positions absolues et les positions relatives des caméras.

Une autre perspective consiste à utiliser de manière plus poussée la modélisation des réseaux urbains proposée pour améliorer la recherche, par exemple si des événements se passent sur des lignes de réseaux de transport.

Bibliographie

Ay S. A., Kim S. H., Zimmermann R. (2010). Generating synthetic meta-data for georeferenced video management. In *Proceedings of the 18th sigspatial international conference on advances in geographic information systems*, p. 280–289. New York, NY, USA, ACM. Consulté sur http://doi.acm.org/10.1145/1869790.1869830

Brinkhoff T. (2002, jun). A framework for generating network-based moving objects. *Geoinformatica*, vol. 6, n° 2, p. 153–180. Consulté sur http://dx.doi.org/10.1023/A:1015231126594

Cucchiara R. (2005). Multimedia surveillance systems. In *Proceedings of the third acm international workshop on video surveillance and sensor networks*, p. 3–10. New York, NY, USA, ACM.

Debnath H., Borcea C. (2013). Tagpix: Automatic real-time landscape photo tagging for smartphones. In *6th international conference on mobile wireless middleware, operating systems, and applications.*

Jensen C. S., Lu H., Yang B. (2009). Graph model based indoor tracking. In *Tenth international conference on mobile data management: Systems, services and middleware, 2009*, p. 122-131.

Keval H. U. (2009). *Effective design, configuration, and use of digital cctv*. Thèse de doctorat non publiée, University College London.

Liu K., Li Y., He F., Xu J., Ding Z. (2012). Effective map-matching on the most simplified road network. *In Proceedings of the 20th International Conference on Advances in Geographic Information Systems*, p. 609-612.

Liu X., Corner M., Shenoy P. (2009). Seva: Sensor-enhanced video annotation. *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 5, n° 3, p. 1–26.

Sandu P. I., Zeitouni K., Oria V., Barth D., Vial S. (2011). Indexing in-network trajectory flows. *The VLDB Journal*, vol. 20, n° 5, p. 643-669.

Sèdes F., Sulzer J., Marraud D., Mulat C., Cepas B. (2012). Intelligent video surveillance systems. In J.-Y. Dufour (Ed.),, chap. A Posteriori Analysis for Investigative Purposes. Wiley.

Shahabi C., Banaei-Kashani F., Khoshgozaran A., Nocera L., Xing S. (2010). Geodec: A framework to visualize and query geospatial data for decision-making. *IEEE Multimedia*, vol. 17, n° 3, p. 14-23.

Shen Z., Arslan Ay S., Kim S. H., Zimmermann R. (2011). Automatic tag generation and ranking for sensor-rich outdoor videos. In *Proceedings of the 19th acm international conference on multimedia*, p. 93–102. New York, NY, USA, ACM.