

# Reconstruction d'animation 3D de rivières à l'aide d'une vidéo

Barros Mikel-Ange – Oran Mikail

21/05/2020

## Résumé

Ce document présente la création d'une application permettant la modélisation d'une rivière à partir d'une vidéo. L'objectif de ce projet est principalement de mettre en place de nouvelles méthodes d'analyse de rivière dans le but de l'utiliser dans les domaines du jeu vidéo ou du dessin animé. En effet, actuellement quand on modélise une rivière, il est important de devoir placer les bons paramètres aux bons endroits de notre rivière afin qu'elle ait un rendu réaliste. La recherche de tels paramètres peut être coûteuse en temps et en énergie. Grâce à la méthode présentée ici, la récupération de ces différents paramètres pourrait se faire de manière beaucoup plus rapide et automatique.

### Table des matières

<b>I) Remerciements</b>	2
<b>II) Présentation du Projet</b>	2
<b>II.1) Introduction</b>	2
<b>II.2) Le contexte</b>	2
<b>II.3) Etat de l'Art</b>	2
II.3.a) Analyse d'image	2
II.3.b) Modélisation 3D	2
<b>II.4) Outils Utilisés</b>	3
<b>III) Méthodologie et mise en place</b>	3
<b>III.1) Présentation de la méthode</b>	3
<b>III.2) Découpage de la rivière</b>	3
III.2.a) Analyse du mouvement	3
III.2.b) Découpage en zone et segmentation	4
III.2.c) Fusion des résultats	4
III.2.d) Liste de vidéos et des résultats obtenus	5
<b>III.3) Détection des différentes zones de rivières</b>	5
III.3.a) Extraction d'informations	6
III.3.b) Détection et découpage des différentes zones	8
<b>III.4) Modélisation de la rivière</b>	8
<b>III.5) Limite de notre implémentation</b>	9
<b>IV) Résultats</b>	9
<b>V) Conclusions</b>	10
<b>VI) Références</b>	10
<b>VII) Annexe</b>	11
<b>VII.1) Pipeline d'exécution</b>	11
<b>VII.2) Manuel d'utilisation</b>	13

## I) Remerciements

Avant de commencer nous tenons à remercier M. A. Peytavie notre Professeur référant sur le projet qui nous a bien aidé et permis de réaliser ce projet.

Nous tenons aussi à remercier M. E. Galin, M. A. Peytavie, M. T. Dupont, M. E. Guerin, M. Y. Cortial, M. B. Benes et M. J. Gain qui nous ont permis d'utiliser une partie de leur travail afin de calculer les déplacements de la rivière lors de la modélisation 3D.

## II) Présentation du Projet

### II.1) Introduction

Dans le cadre de notre master 1 et dans le but d'intégrer le master 2 ID3D il nous a été demandé de réaliser un projet sur un sujet libre proposé par un professeur et nous avons choisi de créer une application permettant à partir d'une vidéo de rivière de créer une représentation réaliste de la rivière à l'aide d'un maillage 3D. L'objectif de ce projet est de mettre en avant une nouvelle méthode, plus simple et rapide, pour modéliser une rivière animée dans un jeu vidéo, un film ou tout autres décors 3D. La méthode utilisée ici, sera une méthode dite à extraction de primitive. Une méthode à extraction de primitive consiste à découper une vidéo en zone et à extraire des caractéristiques qui semblent pertinentes pour la modélisation future.

### II.2) Le contexte

Pour donner suite à la lecture de l'article [NX19], il a semblé à notre professeur qu'une méthode différente pour reproduire une rivière à partir d'une vidéo existait. En effet, bien que donnant d'excellents résultats, la méthode d'analyse SPH nous a semblé couteuse en temps et en ressources et nous pensions pouvoir atteindre des résultats similaires avec une méthode d'extraction de primitive. C'est en partant de ce constat, que nous avons décidé de mettre en place le modèle que nous vous présentons maintenant.

### II.3) Etat de l'Art

#### II.3.a) Analyse d'image

L'analyse d'image est un domaine ayant vu sa popularité exploser sur les dernières années étant donné son intérêt pour extraire des informations de scènes ou d'objet depuis une vidéo ou une photo. Ainsi on peut la voir utiliser sur des téléphones ou robots afin d'identifier leur environnement. C'est dans cette optique que de nombreuses méthodes pour extraire et identifier une rivière sur une vidéo sont apparues. On peut notamment citer les articles de Pedro Santana [PS12] et de René Vidal [RV05] qui nous ont permis d'apprendre et de mettre au point notre propre méthode d'analyse.

#### II.3.b) Modélisation 3D

La modélisation de maillage et leur animation sont des disciplines récentes trouvant de nombreuses applications que ce soit dans le monde du jeu vidéo, du dessin animé ou encore de la médecine. Ainsi, le domaine fait l'objet de nombreuses recherches et notamment dans le but d'automatiser la génération d'objets et de terrains réaliste. En ce qui concerne la génération automatique de rivière de nombreux travaux existent. Et, bien que basés sur des méthodes différentes ces travaux sont complémentaires et ont permis des avancées intéressantes dans la génération de rivières réalistes. Parmi ces nombreux travaux nous pouvons citer celui d'A. Peytavie [AP19] et celui de P. Nugjar [PN13] qui, bien que basés sur deux méthodes différentes, nous ont tous les deux servi dans la mise en place de nos résultats et de notre modèle.

## II.4) Outils Utilisés

Afin de mettre en place cette application, nous utiliserons deux outils :

- L'application « QT Creator » ainsi que les bibliothèques associées afin de créer l'application et de mettre en place l'affichage 3D
- La bibliothèque OpenCV afin d'analyser la vidéo.

Notre algorithme sera testé sur des vidéos au format 1280\*720 provenant de la plateforme YouTube.

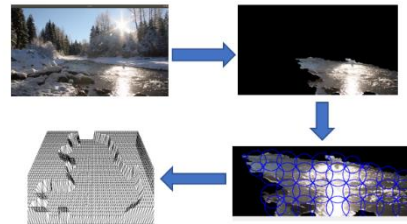
## III) Méthodologie et mise en place

### III.1) Présentation de la méthode

Le programme mis en place se découpe en 3 grandes parties, comme vous pouvez le voir sur la pipeline (0) :

- L'identification de la rivière
- Le clustering de la rivière
- La modélisation de la rivière

Chacune de ses trois grandes parties sera expliquée plus en détail dans les parties qui suivent



(0) Pipeline basique

### III.2) Découpage de la rivière

Le premier objectif de ce projet a été d'analyser la vidéo et d'essayer de trouver comment découper la forme de la rivière du reste du décor. Pour cela, nous avons passé les premiers jours à regarder des vidéos sur des rivières afin d'en retirer des points communs utiles à notre analyse. En ce sens, deux points nous ont sauté aux yeux. Premièrement, une rivière est en mouvement et cela peut être utilisé. Deuxièmement, une rivière a une colorimétrie particulière qui peut être étudiée.

#### III.2.a) Analyse du mouvement

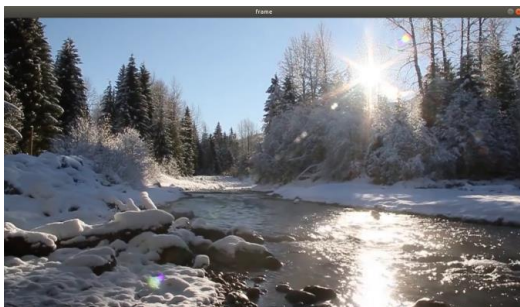
Une rivière est un élément en perpétuel mouvement, ce qui est rarement le cas du reste d'une scène. Et c'est pourquoi lors de notre analyse nous partons du principe qu'une rivière est le seul élément en mouvement sur la vidéo. C'est ainsi que nous avons décidé d'analyser le mouvement de notre vidéo en nous basant sur la méthode de l'article [PS12].

Ainsi nous effectuons une détection du mouvement à l'aide de l'algorithme de Lucas Kanade. [LKOC]

L'algorithme de Lucas Kanade consiste à placer un certain nombre de points significatifs sur notre image et d'en suivre le déplacement.

Par la suite, nous traitons les résultats obtenus afin de calculer l'entropie de chaque point en fonction de ses voisins ainsi que de la distance parcourue par ce point.

Enfin, après avoir calculé l'entropie, nous remplaçons les points à leur position sur l'image de base (1) et nous dessinons le contour de notre zone en mouvement ainsi détectée (2)



(1) image de base à analyser



(2) zone détectée comme étant de la rivière

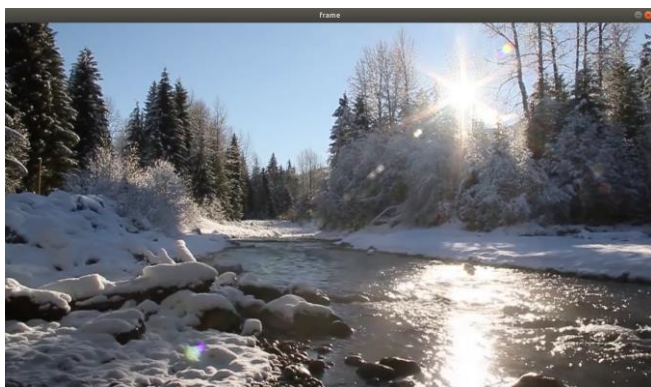
### III.2.b) Découpage en zone et segmentation

Nous avons vite remarqué que l'analyse du mouvement n'était pas suffisante pour avoir notre rivière et c'est pour quoi nous avons décidé d'analyser, en plus du mouvement, la colorimétrie de la zone. Pour cela nous avons utilisé 2 algorithmes majeurs, l'algorithme des K-means [KMOC] et le watershed algorithm. [WSOC]

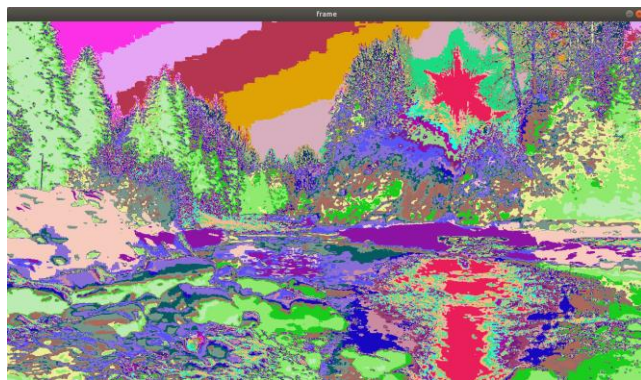
L'algorithme des K-means consiste à placer une série de points d'intérêt sur l'image et à faire grossir des clusters en minimisant la distance entre la couleur du pixel centrale au cluster et la couleur des autres pixels du cluster. Le watershed algorithm quant à lui sert à regrouper les zones similaires entre elles. L'algorithme du watershed n'est pas forcément le plus performant, et un algorithme personnalisé aurait pu donner de meilleurs résultats. Mais comme les résultats obtenus avec cette méthode étaient plus que satisfaisant, nous avons décidé de garder cette méthode simple et peu couteuse.

L'exécution de notre programme s'effectue dans ce sens, nous passons une première fois notre image de base (3) dans un algorithme des K-means avec 32 clusters (4). Nous obtenons ainsi de nouvelles zones définies avec des couleurs aléatoires. Ensuite nous faisons passer l'image ainsi obtenue dans un deuxième algorithme des K-means mais cette fois avec seulement 16 clusters (5).

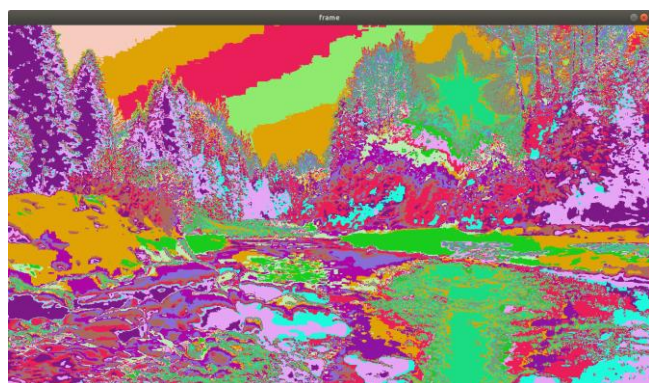
Et enfin nous le faisons passer dans le watershed algorithm pour découper nos zones finales (6).



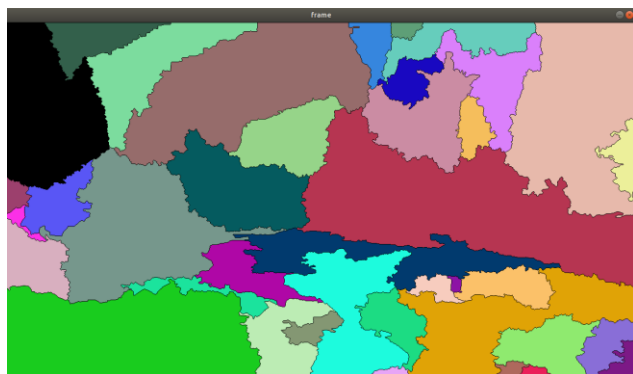
(3) l'image de base



(4) l'image en sortie du K-means 32 clusters



(5) l'image en sortie du K-means 16 clusters



(6) l'image en sortie du Watershed

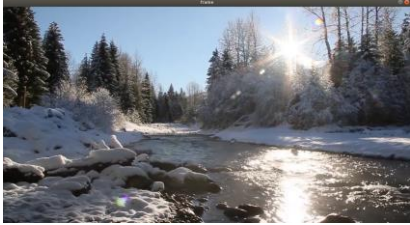







### III.2.c) Fusion des résultats

Enfin et pour conclure sur cette partie nous avons dû rassembler les résultats entre notre détection de mouvements et notre détection de zones. Pour cela, nous regardons si pour chaque zone de notre image



segmentée nous avons plus de la moitié des pixels définis comme étant de la rivière. Si c'est le cas nous considérons que l'entièreté de la zone correspond à une rivière et nous l'ajoutons au résultat final. Ainsi nous obtenons une assez bonne précision sur les contours de notre rivière et nous pouvons passer à l'analyse des différentes zones composant la rivière.

### III.2.d) Liste de vidéos et des résultats obtenus

Images de départ	Zones découpées	Nombre de frames/durées d'exécution
		1500/1min05s
		1500/1min05s
		1300/55s
		1500/1min05s

### III.3) Détection des différentes zones de rivières

Après avoir défini la zone de notre rivière, il nous restait un problème à régler : une rivière ne se comporte pas de la même manière partout et nous ne pouvions donc pas définir un ensemble de paramètres globaux sur notre rivière. La question c'est alors posé, Comment allons-nous faire pour extraire les informations des différentes zones de notre rivière ?

Pour répondre à cette question nous avons procédé en deux étapes. Tout d'abord, nous avons choisi d'extraire des informations sur un nombre de points limité afin de comprendre et analyser le comportement de chaque

zone. Une fois cette première analyse terminée, nous découpons la rivière en zone de taille fixe et nous récupérons les informations des différents indicateurs pour chaque zone.

III.3.a) Extraction d'informations

Dans cette partie, nous avons donc réfléchi au comportement d'un groupe de point pendant l'analyse et aux informations que nous pouvions extraire grâce à une vidéo. Ainsi, trois indicateurs nous ont semblé importants, la vitesse, la colorimétrie et le sens de déplacement des points. Et c'est pourquoi, pour pouvoir étudier le comportement des différents indicateurs nous avons mis au point 4 méthodes de suivi des points. Nous pouvons ainsi suivre un seul point, un cluster de point, un point lancé 20 fois à la même position avec une image de décalage avant d'analyser son mouvement ou sur l'ensemble des points d'intérêts de l'image.

III.3.a.1) Suivi d'un point unique

Dans cette première méthode nous avons créé différents graphiques pour chaque point afin d'avoir une analyse plus visuelle des données et de pouvoir les comparer.


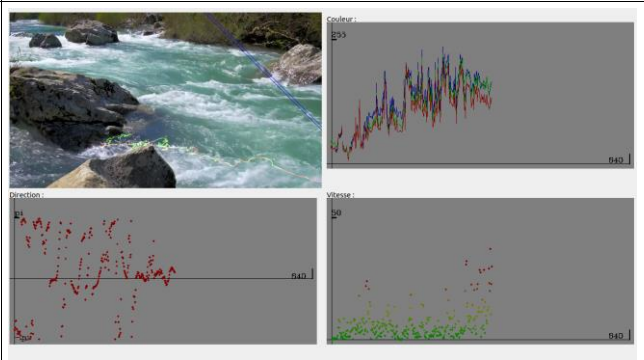
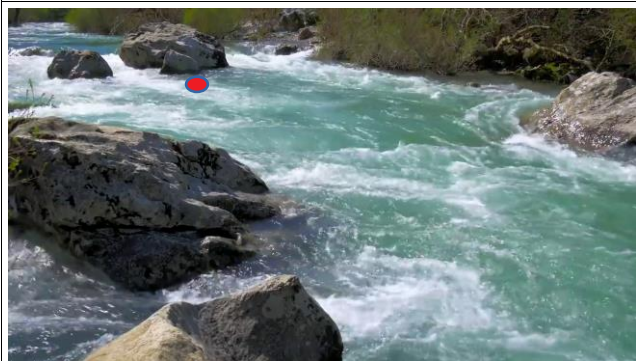
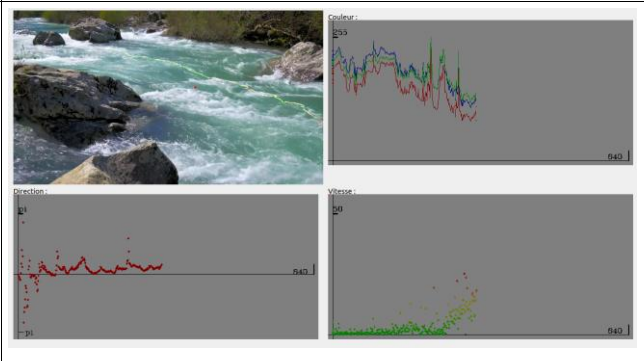
Le premier graphique (celui de couleurs) comporte trois courbes correspondant aux trois composantes RGB de la couleur de l'image pouvant varier chacune entre 0 et 255.

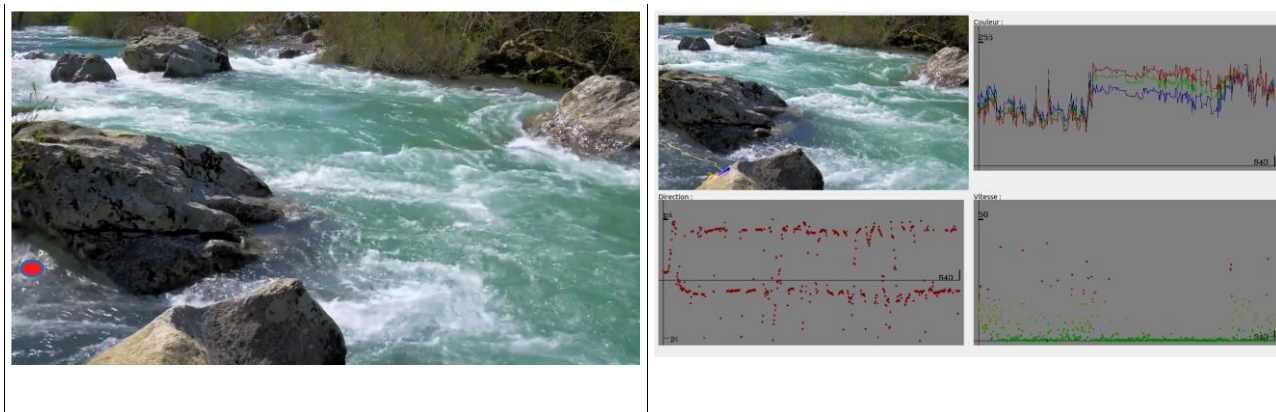
Le deuxième graphique (celui d'orientation) comporte une courbe correspondant à l'angle de la courbe calculée à partir de son cosinus et comprise entre  $-\pi$  et  $\pi$

Le troisième graphique (celui de vitesse) comporte une courbe correspondant à la distance euclidienne parcourue par le point toutes les 10 frames.

Ainsi cette méthode nous a permis de mettre en évidence les différences de comportement des zones de notre rivière et d'afficher les informations qui nous semblaient pertinentes afin d'extraire des caractéristiques communes à notre rivière pour chaque zone.

En voici, quelques exemples :

Position de départ du points	résultats
	
	



### III.3.a.2) Suivi d'un cluster de points

Ici (7a) l'objectif est de déterminer pour quelle taille de zone les points se comportent de la même manière. Pour cela nous envoyons un ensemble très important de points définis sur une zone de taille fixe et nous regardons le comportement de chaque point sur cette zone. Grâce à cela nous avons pu déterminer qu'une zone de taille 80\*80 pixels était le plus optimal pour éviter les erreurs.



(7a) mouvement d'un cluster

### III.3.a.3) Suivi d'une série de points

Avec ce mode (7b) nous avons essayé de déterminer si une direction commune était prise par un même point quel que soit le moment auquel il partait et si ainsi nous pouvions éliminer un certain nombre d'erreurs de direction dans le calcul de mouvement de notre rivière. L'analyse des résultats nous a faits remarquer que notre algorithme de détection du mouvement détectait un certain nombre de mouvements, inutiles pour déterminer la direction principale et que nous pourrions donc extraire un certain nombre d'erreur pour avoir une direction plus précise dans notre modélisation.



(7b) mouvement d'une série de points

### III.3.a.4) Suivi de l'ensemble des points

Ici, l'analyse s'effectue au milieu de la boucle principale d'exécution. Ainsi, nous pouvons analyser la cohérence de nos résultats sur la rivière entière. Pour cela, nous avons donc choisi d'étudier l'ensemble des points d'intérêts sur notre image et pour cela nous affichons en temps réels la position, la vitesse et la direction de chacun de nos points.

Pour que cette représentation soit visuelle nous avons affiché nos résultats selon le code suivant : Plus la vitesse est importante, plus la couleur du point tendra vers le rouge et à l'inverse, plus la vitesse sera faible, plus la couleur du point tendra vers le vert.

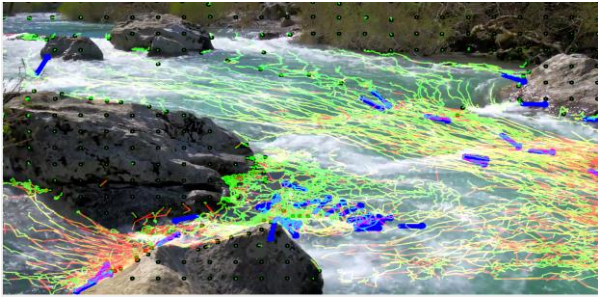
Une flèche orientée indique la direction de déplacement de notre point.

Ici la vitesse est définie comme étant la distance euclidienne entre la position de chaque point à l'image n et l'image n-1.

L'orientation est quant à elle définie comme étant un point à deux dimensions composé du déplacement normalisé en X et du déplacement normalisé en Y du point entre l'image n et l'image n-10.

On peut ainsi constater la cohérence de nos résultats (8) et passer à la suite, l'analyse par cluster.





(8) résultat de l'analyse

### III.3.b) Détection et découpage des différentes zones

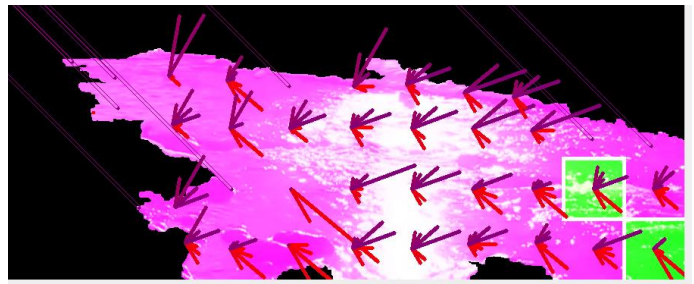
Une fois ces différentes analyses effectuées nous pouvons revenir sur le mode principal et commencer à analyser les zones une à une. On découpe donc notre rivière selon une grille de carrés de 80 pixels\*80 pixels.

Une fois cette découpe faite, nous faisons passer notre algorithme de Lucas Kanade sur l'image de base. Et nous détectons tous les points passant par la zone découpée. Pour chaque zone on récupère la moyenne de la vitesse de déplacement de tous les points dans la zone, la moyenne de colorimétrie de la zone et la direction prise par chaque point de la zone selon une rosace de direction à 16 valeurs.

A partir de cela on détermine les zones composants notre rivière selon 5 types :

- Calme (rose)
- Calme avec remous (vert)
- Rapide (bleu)
- Rapide avec remous (jaune)
- Tourbillon(rouge)

Et l'on crée notre image résultat (9)



(9) Image coupée par type

### III.4) Modélisation de la rivière

Pour modéliser la rivière nous avons utilisé le code mis en place dans l'article [AP19] et mis à notre disposition par notre professeur de POM. La première chose à faire est de modifier notre représentation des zones. En effet, nous utilisons de base un découpage en rectangle mais le code envoyé par notre professeur utilisait lui des zones circulaires. Pour arriver à ce résultat nous avons calculé pour chaque zone le cercle le plus petit entourant ladite zone (10).

Maintenant que nous avons nos zones, nous pourrions modéliser la rivière mais il nous reste un détail à régler. Nous devons redéfinir la perspective, afin d'obtenir une rivière vue de dessus dans notre modélisation 3D. Pour cela, nous avons décidé de laisser la main à l'utilisateur. En effet, sur un décor naturel la détection automatique des points de fuites semblait impossible, surtout sans connaître la position de la caméra à la base. Et c'est pourquoi nous avons pris le parti de faire dessiner un trapèze par l'utilisateur (11). Ce trapèze va nous servir de base pour définir notre transformation et c'est pourquoi il doit respecter la forme de la rivière le plus possible.

Une fois le trapèze dessiné, l'utilisateur doit relancer l'exécution et nous calculons les matrices de transformation qui permettent de passer du trapèze à un rectangle de dimension (600,450). Une fois ces matrices obtenues, nous calculons le projeté des points du plan du rectangle sur le trapèze et nous associons chaque point à une zone. De cette manière, nous pouvons déjà modéliser notre rivière en 3D mais il nous manque l'animation. Et c'est là que le code envoyé par M. A. Peytavie nous sera utile. En effet, en créant des zones définies selon son code nous pouvons calculer le déplacement de la rivière. Pour cela, il



nous faut projeter le centre de chaque zone ainsi que la direction de chaque zone du repère du trapèze vers le repère du rectangle.

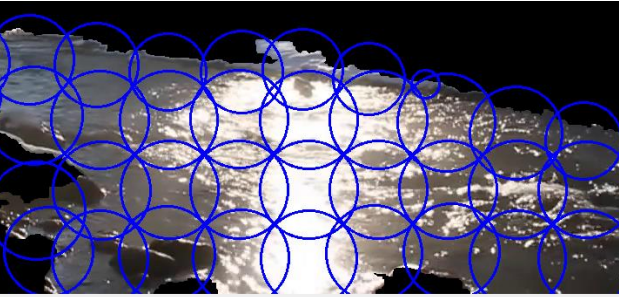
Une fois toutes ces informations obtenues, nous pouvons générer une carte des hauteurs ce qui nous permettra de modéliser notre rivière finale (12). Une carte des hauteurs est une image en nuances de gris qui permet de déterminer la hauteur de chaque point dans un repère 3D.

Les paramètres retenus pour définir le mouvement des rivières sont :

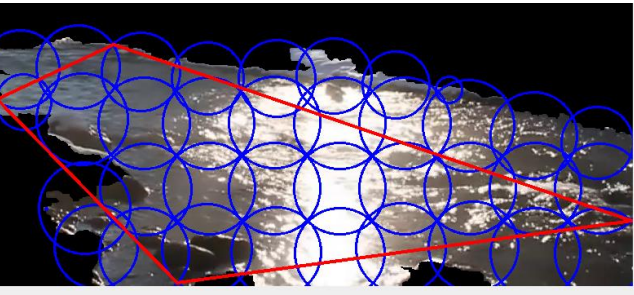
Hauteur des vagues = colorimétrie/255

Largeur des vagues = largeur de la zone/2

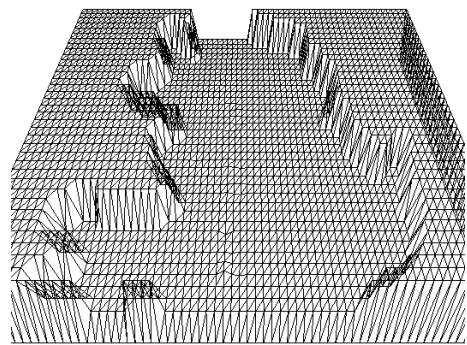
Vitesses de déplacements = vecteur direction\*Vitesse



(10) Zone redéfinis en cercles



(11) dessin du trapèze



(12) Résultat final

### III.5) Limite de notre implémentation

Notre implémentation, bien qu'efficace à deux problèmes majeurs :

Premièrement, elle est dépendante des paramètres passés par l'utilisateur. En effet en fonction du nombre de frame passé, de la forme du trapèzes dessiné les résultats peuvent être très différents.

Deuxièmement, chaque vidéo à besoin de paramètres différents et ce notamment sur le nombre de frame nécessaire à l'exécution.

## IV) Résultats

Dans le tableau, ci-dessous :

T1 : représente le temps pour découper la rivière

T2 : représente le temps pour découper les zones

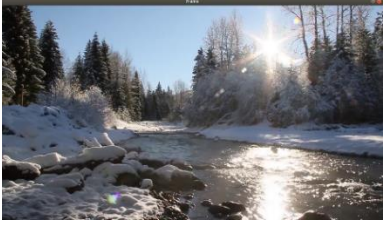
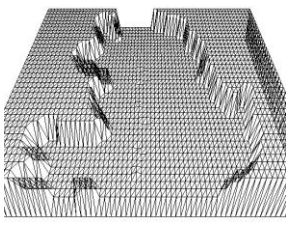

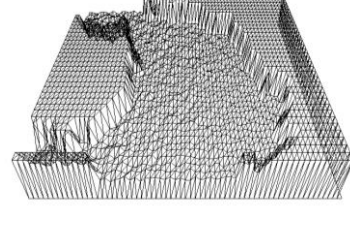

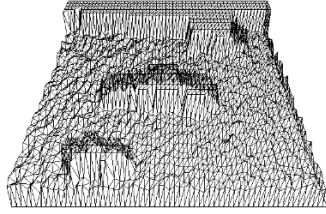
T3 : représente le temps pour faire le rendu 3D

La valeur 1 : représente le temps d'exécution avec affichage de la simulation

La valeur 2 : représente le temps d'exécution sans affichage de la simulation

L'ensemble des résultats ont été obtenus sur un ordinateur équipé d'un processeur ryzen 7 3750H, 4core-8thread et de 16 go de ram

Pour toutes les simulations un clustering en rectangle est utilisé.

Paramètres d'exécution	Rivières de base	T1 (s)	T2 (s)	T3 (s)	Résultats	Temps Total (s)
Nombre de Frames =1500 Taille cluster =80*80		1 : 60 2 : 25	22	2		1 : 84 2 : 49
Nombre de Frames =1500 Taille cluster =80*80		1 : 63 2 : 29	84	3		1 : 150 2 : 116
Nombre de Frames =1500 Taille cluster =80*80		1 : 60 2 : 31	110	2		1 : 162 2 : 143

## V) Conclusions

Durant ce projet, nous avons effectué un travail de recherche visant à définir et mettre en place une nouvelle méthode pour la modélisation automatisée de rivière. C'est dans ce but que nous avons mis en place différents mode d'analyse de vidéo ainsi qu'un mode principal servant à la génération de la rivière. Ce mode principal permet l'analyse, la découpe et la modélisation d'une rivière à partir d'une vidéo selon les paramètres passés par l'utilisateur. Actuellement nous utilisons les travaux de M. A. Peytavié pour générer le modèle final, une prochaine étape de notre projet serait de mettre en place notre propre modèle de génération et d'affichage.

Ce projet nous a permis d'en apprendre beaucoup sur l'analyse d'image et la modélisation 3D. Il nous a aussi permis de mener à bien un vrai travail de recherche et d'analyse de problèmes.

## VI) Références

- [NX19] <https://diglib.eg.org/handle/10.2312/pg20191337>  
[PS12] [http://home.iscte-iul.pt/~pfsao/papers/robio\\_2012.pdf?fbclid=IwAR2ZzSU\\_xjrzuOMv4eE1YNNh4CnxikhhOGxd9SC3yA-pxhVgRt452CiKQPk](http://home.iscte-iul.pt/~pfsao/papers/robio_2012.pdf?fbclid=IwAR2ZzSU_xjrzuOMv4eE1YNNh4CnxikhhOGxd9SC3yA-pxhVgRt452CiKQPk)  
[RV05] <http://www.vision.jhu.edu/assets/VidalCVPR05.pdf>  
[AP19] <https://hal.archives-ouvertes.fr/hal-02281637/file/main.pdf>  
[PN13] [https://www.researchgate.net/publication/255918841\\_Markov-Type\\_Vector\\_Field\\_for\\_endless\\_surface\\_animation\\_of\\_water\\_stream](https://www.researchgate.net/publication/255918841_Markov-Type_Vector_Field_for_endless_surface_animation_of_water_stream)  
[LKOC] [https://docs.opencv.org/3.4/d4/dee/tutorial\\_optical\\_flow.html](https://docs.opencv.org/3.4/d4/dee/tutorial_optical_flow.html)  
[KMOC] <https://docs.opencv.org/2.4/modules/core/doc/clustering.html>  
[WSOC] [https://docs.opencv.org/master/d3/db4/tutorial\\_py\\_watershed.html](https://docs.opencv.org/master/d3/db4/tutorial_py_watershed.html)

VII) Annexe

VII.1) Pipeline d'exécution

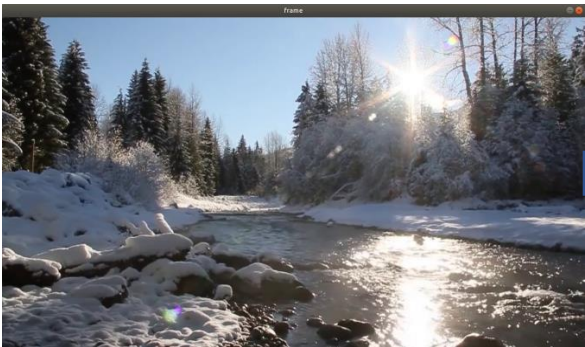


Figure 1 Image de départ

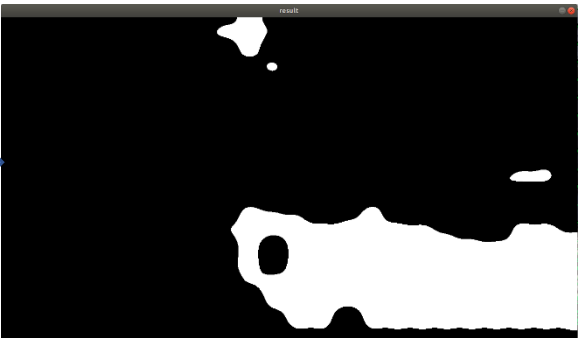


Figure 2 Analyse de la vitesse

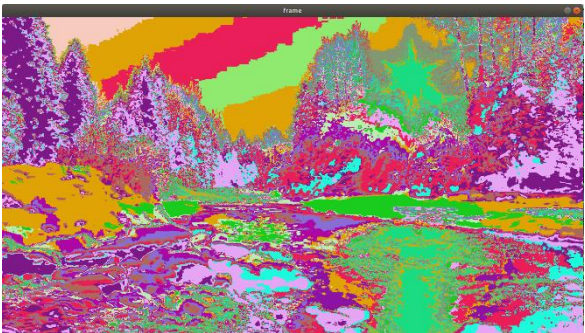


Figure 4 Deuxième K-means

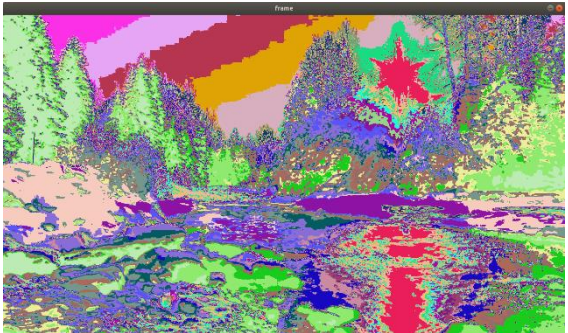


Figure 3 Premier K-means

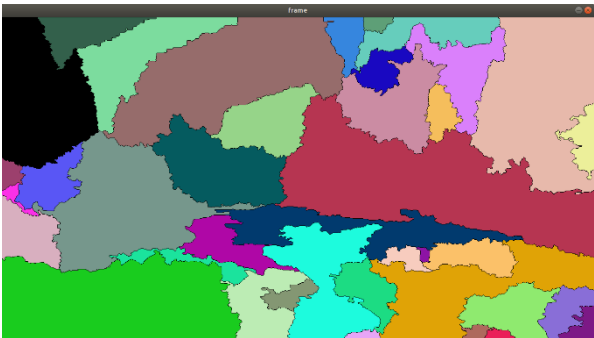


Figure 5 Wattershed

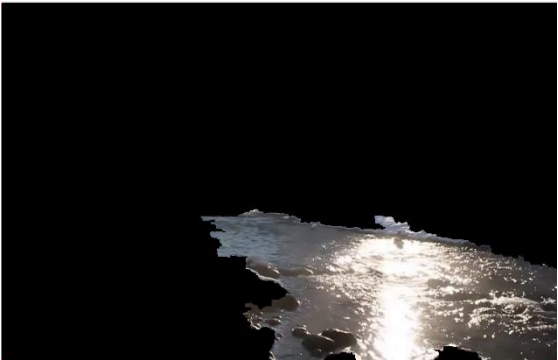
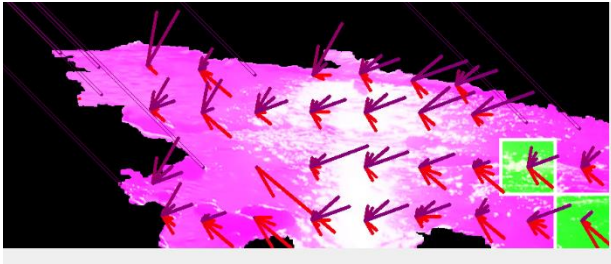
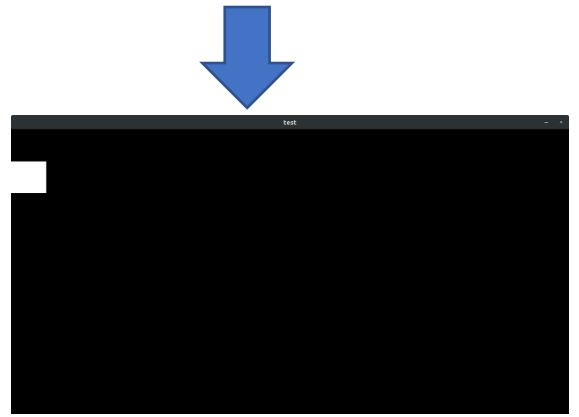


Figure 6 Rivière découpée

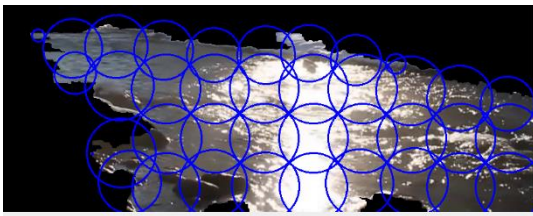




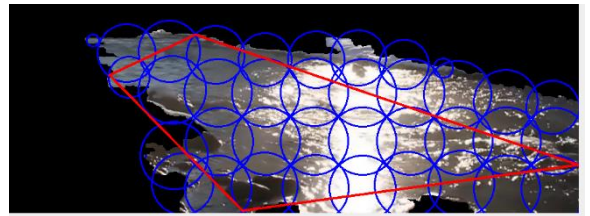
*Figure 8 analyse des informations sur chaque zone*



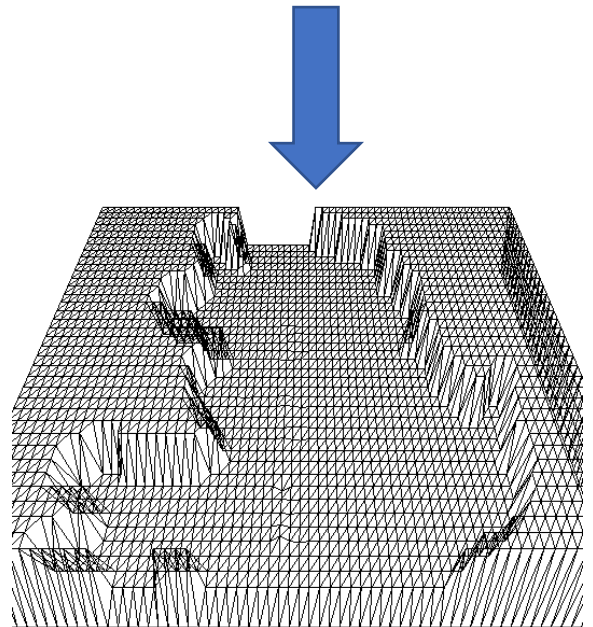
*Figure 7 Définition des zones de rivières*



*Figure 9 transformation des zones rectangulaire en zone circulaires*



*Figure 10 dessin tu trapèze pour la transformation*



*Figure 11 résultat final*

## VII.2) Manuel d'utilisation

Permet de changer la vidéo

Sélectionne le nombre de frame sur lesquelles le programme va s'exécuter

Défini la taille des clusters utilisés pour le découpage de la rivière

Sélectionne le mode d'exécution du programme

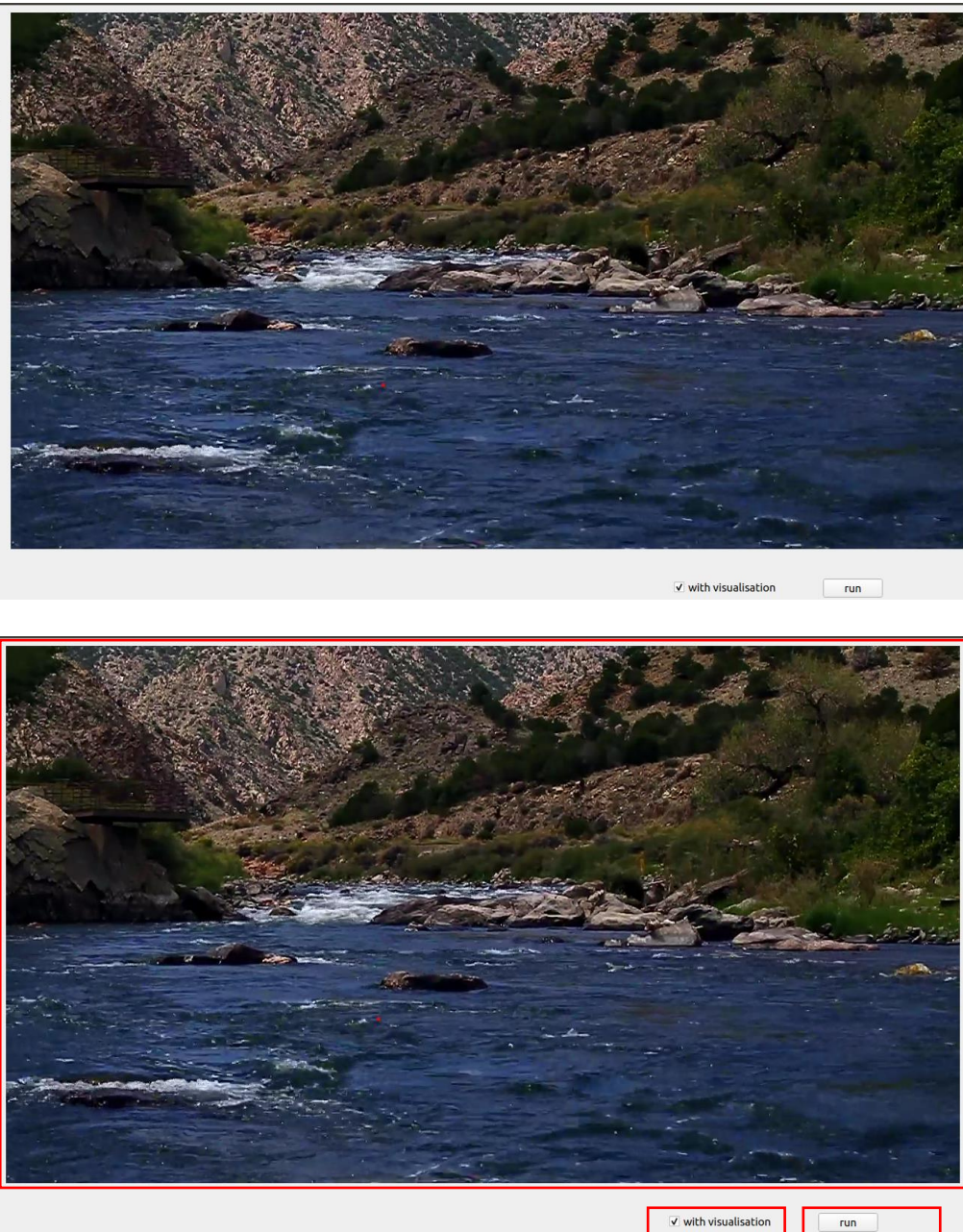
Défini si les points d'intérêts se déplacent sur l'image

Sélectionne le mode de découpage de l'image

Zone d'interaction pour la pose de points d'intérêts Et le rendu final

Défini si l'on aura un aperçu de l'exécution du programme

Lance l'exécution du programme



The screenshot displays a software interface for video analysis. On the left, there is a control panel with the following elements: a 'change video' button, an 'IMAGE ANALYSIS' section with a 'select frame number' dropdown set to '640', a 'select detection mode' dropdown set to 'following one Point', a checked 'Point movement' checkbox, a 'select Size of clustering' section with 'X: 80' and 'Y: 80' input fields, a checked 'zoning with rectangle' checkbox, and a '3D MODELING' section. The main area shows a video frame of a river with a red point and a green rectangle. At the bottom right, there is a 'with visualisation' checkbox and a 'run' button. Red arrows point from the text boxes to these specific UI elements.

Différents modes d'exécutions :

- générale : identification et modélisation de la rivière
- suivi d'un point
- suivi d'un cluster
- suivi d'un point sur 20 frames

Différents modes de clustering :

- en rectangle
- selon un watershed