From photogrammetric reconstruction to immersive VR environment

Maxime Lhuillier

Université Clermont Auvergne, CNRS, Institut Pascal, F-63000
Clermont Ferrand, France

IC3D, 8th december 2021, Brussels, Belgium

Introduction

Steps to generate a VR environment from images of a real scene

(1) Choose experimental conditions and take the images

(2) Reconstruct a textured 3D model thanks to a photogrammetry method

(3) Import the 3D model into a game engine to generate a VR application

Motivations

VR needs a 3D model in a coordinate system that has a physically plausible scale and a vertical coordinate axis

However standard photogrammetry does not provide this

How to choose such a coordinate system after (2) and before (3) ?

Introduction

VR needs

A good scale for stereoscopic rendering, realistic speed and motions of the user in VR environment

The vertical direction must be known for drawing as vertical lines in the HMD vertical lines in 3D, when the user looks the scene with usual poses of the head (no roll, no pitch)

Otherwise the VR experience is unrealist and cybersickness can increase

Definition of a « good » coordinate system of the 3D model

The 3D coordinates of the vertices are in meters

A coordinate axis (Oz in the paper) is vertical and points toward the sky

Previous work

Photogrammetry with only a camera sensor

The coordinate system is set by structure-from-motion (SfM)

If the sensor has only one centre, SfM can only reconstruct up to a similarity transform of the 3D space (arbitrary rotation-translation-scale)

If it has several centres (eg a multi-camera) with a known distance, SfM can only reconstruct up to an Euclidean transform (arbitrary rotation-translation)

Thus the coordinate system has no chance to be a good one [Hartley 2000]

In practice

Manually choose the coordinate system with a GUI (eg Blender, Unity ...), or

Compute the vertical assuming that the camera motion is planar & horizontal

Previous work

Photogrammetry with additional sensor(s)

GPS and IMU measures can be integrated in SfM [Klingner 2013 ...]

GPS provides georeferenced reconstruction, thus scale and vertical are known

IMU provides the vertical if it does not move, and scale by integrating over time

Each sensor has drawback: unavailable GPS in urban canyons, drift for IMU

Paper contribution

Reset the coordinate system of photogrammetry such that it becomes « good »

Do not need an additional sensor

Do not assume that the camera motion (acquisition) is planar and horizontal

Assumptions

Experimental context

360 camera moving on the ground for complete and convenient reconstruction

Complete: with ground surface, façades, all kinds of ceilings (foliages, sky, ...)

Convenient: no need to rotate a camera to reconstruct all around

Four assumptions

The camera height (with respect to the ground) is roughly constant and known

The reconstructed surface is closed (it also covers the ceilings)

The camera height is smaller than the distance between camera and ceiling

Scene anisotropy: the density of surface normals is higher near the horizontal directions and near the vertical direction than elsewhere. (The main scene components are vertical, eg facades, or roughly horizontal, eg ground)

Reset the coordinate system

Notations for unknowns

Let v be the vertical direction pointing toward the sky. This is a unit vector in the 3D coordinate system used and provided by photogrammetry.

The unoriented vertical is e*v where e=-1 or e=+1

Principle

Find a rough estimate v’ of e*v from the camera trajectory using PCA

Estimate e*v by a 2D Hough transform in a search space centered on v’

Find the good e thanks to distance computations between camera and surface

Obtain v and the scale ratio (between new and old coordinate systems) from e

Last update the coordinate system using v and the scale ratio

Reset the coordinate system

Rough estimate v’ (PCA)

Let C be the covariance matrix of the locations of the acquisition camera

Let v’ be the singular vector of the smallest singular value of C

If the camera motion is planar and horizontal, v=-v’ or v=+v’

Search space of e*v

It is a set of unit vectors u forming angle with v’ that is less than a threshold t

A large enough t is needed to deal with roughly planar ground surfaces with moderated slope angles. t< pi/2 since a hemisphere is enough.

Each vertical candidate u corresponds to a pixel of an image by a function

The function is defined by a pinhole camera (FoV=2*t, principal direction=v’)

Reset the coordinate system

2D Hough transform

Every triangle of the surface votes for every vertical candidate u that is parallel to the triangle. Ie triangle normal and u are orthogonal (up to sampling)

A triangle vote is weighted by the triangle area (invariance to mesh subdivision)

Last, e*v is the vertical candidate that maximises the votes

Explanation

The vertical candidates of a triangle are in a great circle of the unit sphere

If a triangle is vertical, its great circle always include the two opposite **vertical** directions, only one is in the search space (if t is large enough)

Thus every vertical triangle provides a vote for the (unoriented) vertical direction

Reset the coordinate system

Bad maximiser of the votes

Every vertical triangle provides a vote for the vertical direction

Every horizontal triangle provides a vote for every horizontal direction

Thus the maximiser can be horizontal if the ground is a main part of the scene !

No other case of bad maximiser thanks to the anisotropy assumption

Avoid bad maximiser thanks to the search size t

The angle between v' and the vertical is smaller than the angle between v' and every horizontal direction (since v' is roughly vertical)

Choose a value of t that separates these two kinds of angles, here t=pi/4

Then the horizontal directions are not in the search space, but the vertical is

Reset the coordinate system

From unoriented to oriented vertical direction

Now e*v is known (e and v are still unknown), thus we know the function e->v(e)

For each e in {-1,+1}, estimate the distance m(e) between the camera trajectory and the triangles that are “below” the camera trajectory in the sense of v(e)

Thanks to the assumptions, m(-1) and m(+1) are camera-ground or camera-ceiling distances, and the smallest one is the camera-ground distance

Thus v=v(e) such that m(e)< m(-e)

Estimation of m(e)

Compute the intersection(s) between the surface and the half-line started at a camera location with direction -v(e)

m(e) is the mean of distances between each location and their intersections

Reset the coordinate system

Update the coordinate system used by photogrammetry

Thanks to the assumptions, the camera height h is known in meters

Vector v becomes (0 0 1) and distance min{ m(-1),m(+1) } becomes h

The coordinates of all surface vertices are multiplied by h/min{ m(-1), m(+1} }

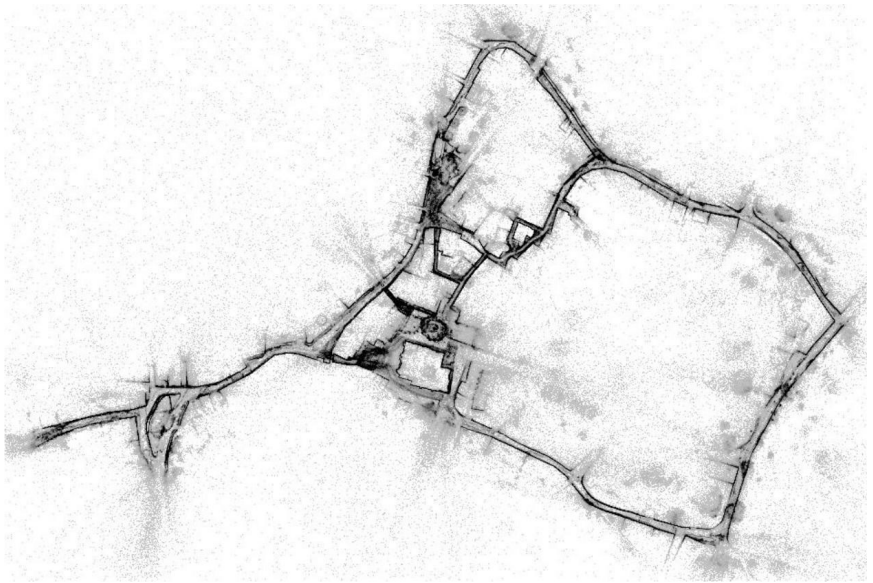
They are also multipled by a rotation that maps v to (0 0 1)

Input images for photogrammetry

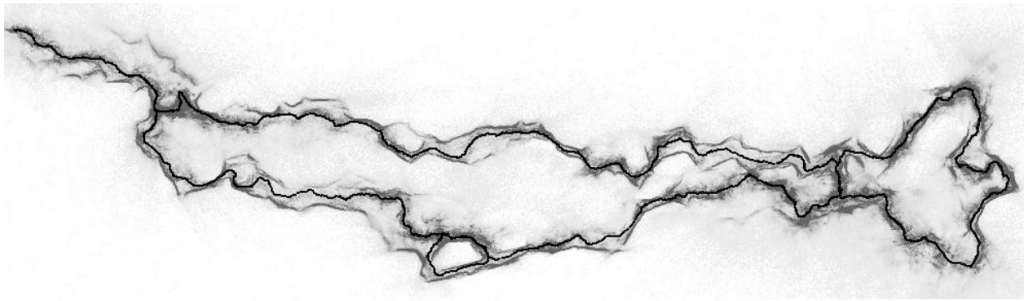
Use a helmet-held Gopro Max 360 camera and obtain a video of equiangular cubemap images at 30Hz (each cube face is 1344*1344)

Convert each eq. cubemap image in a format supported by a photogrammetry software (equirectangular, standard cubemap, or fisheye images)

Dataset



(a) City's points



(b) Basalt's points and camera trajectory

Name	#tri	#loc	h	length	slope-length
Basalt	2.9M	3840	1.8m	819m	>30%,496m
City	3.4M	3132	1.55m	1.6km	>10%,240m

Table 1. Dataset's characteristics: number of triangles, number of selected camera locations $\mathbf{l}_i$, camera-ground distance (i.e. camera height) h, trajectory length, slope-length (e.g. at least 10% slope along 240m of the acquisition trajectory).

[12] M. Lhuillier. Surface reconstruction from a sparse point cloud by enforcing visibility consistency and topology constraints. *Computer Vision and Image Understanding*, 175, 2018.

[13] M. Lhuillier. Local-convexity reinforcement for scene reconstruction from sparse point clouds. In *International Conference on 3D immersion*, 2019.

Comparisons between PCA and Hough for Basalt

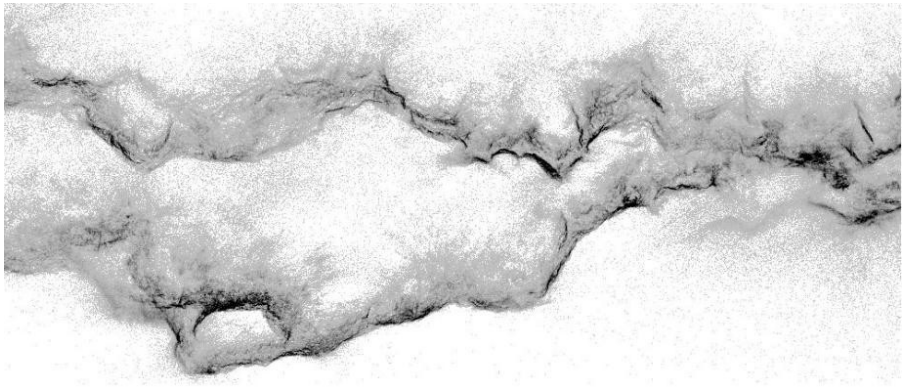
Reminder: v'= PCA and e*v=Hough

Project the cloud of reconstructed points using a projection parallel to a vertical direction

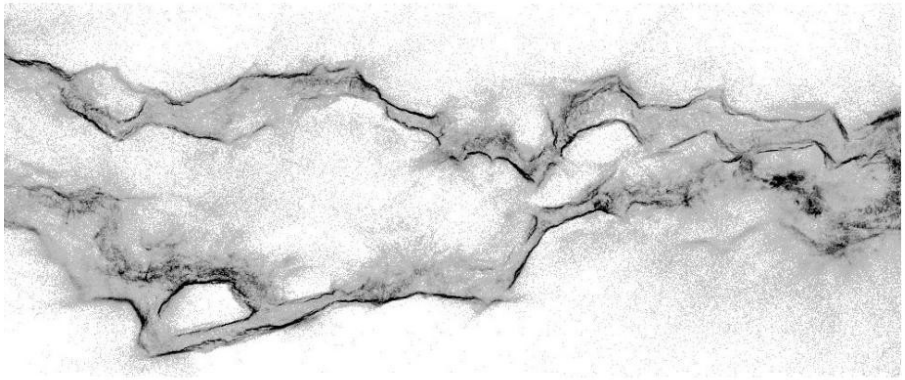
The direction of projection is v' on the top and e*v on the bottom

The vertical scene components (canyon sides) are projected as curves, if the direction of projection is vertical

Thus e*v is better than v'



(a) Projection parallel to the PCA direction ($\mathbf{v}_0$)



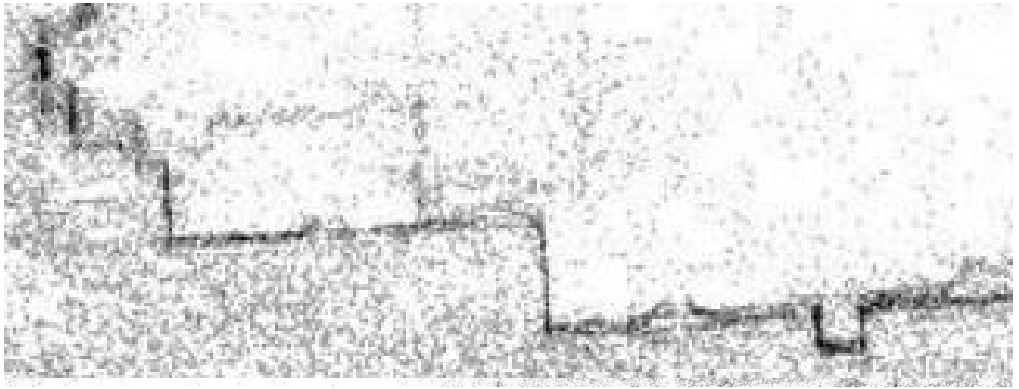
(b) Projection parallel to the Hough direction ($\epsilon\mathbf{v}$)

Comparisons between PCA and Hough for City

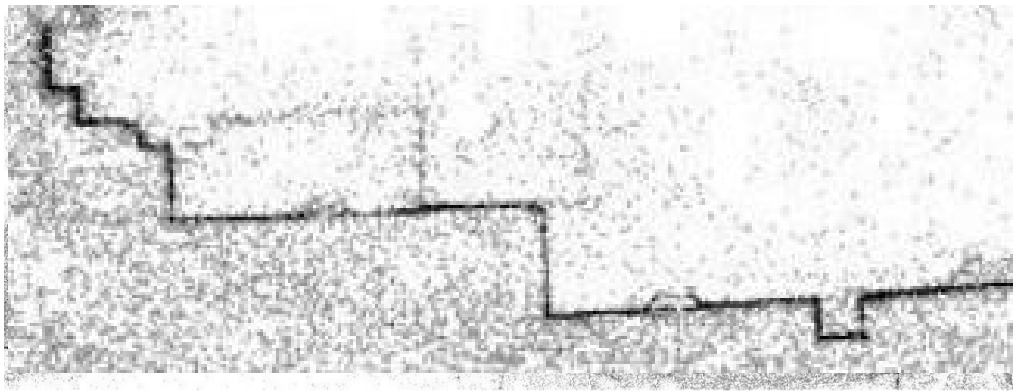
The same comparison for City is less evident: the line segments look similar

A careful observation shows that line segments are less noisy or darker (or have smaller thickness) for e*v than for v'

Thus e*v is better than v'



(a) Projection parallel to the PCA direction ($\mathbf{v}_0$)



(b) Projection parallel to the Hough direction ($\epsilon\mathbf{v}$)

Comparisons between PCA and Hough for City

Draw the 3D model by a pinhole camera that has zero pitch and zero roll with respect to a vertical direction of reference

The direction of reference is v’ on the top and e*v on the bottom

The vertical lines in 3D are drawn as vertical lines in 2D, if the reference direction is accurate

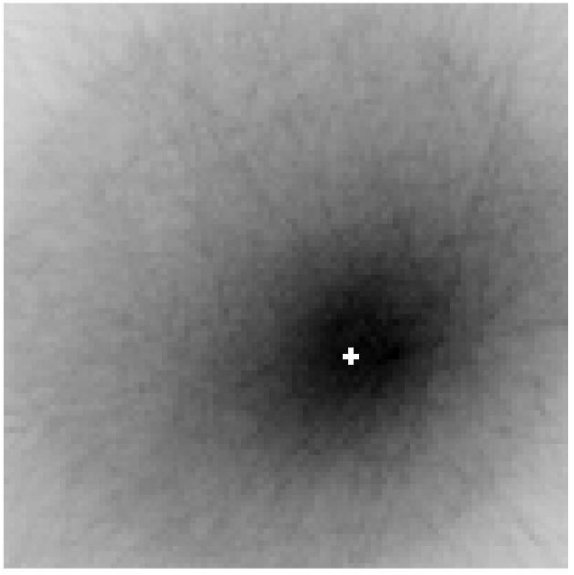
Thus e*v is better than v’

Accumulator spaces of the Hough transform

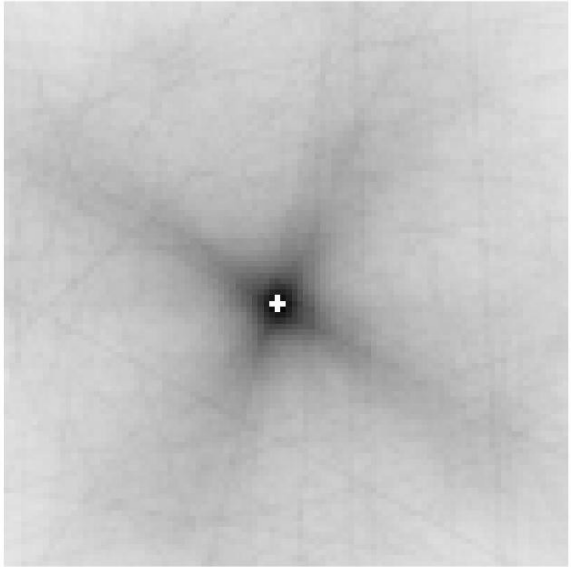
A search space is centered at v' and bounded by angle t=pi/4

The vertical candidates are mapped to a 100*100 image by a pinhole camera

The darker the gray level, the greater the vote



(a) Basalt's accumulator



(b) City's accumulator

VR applications to explore the immersive models

The user of VR has several capabilities

Can move on the ground like a pedestrian (or fly in 3D like a bird)

Can follow the trajectory of the acquisition camera for a naive exploration where the visual quality is good (or go away to see photogrammetry inaccuracies)

The motions of the user are similar to those in previous VR applications by using joysticks (continuous translation and discrete rotation)

Availability

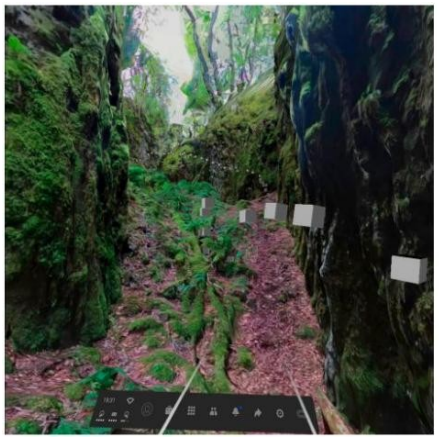
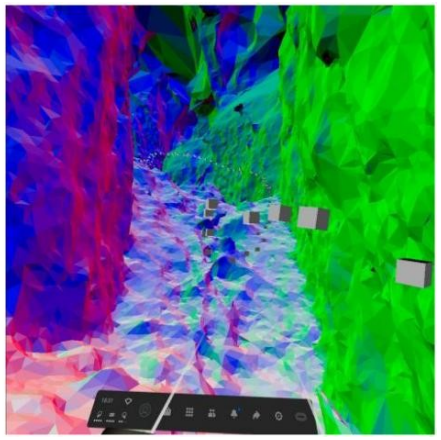
Download the applications from http://maxime.lhuillier.free.fr

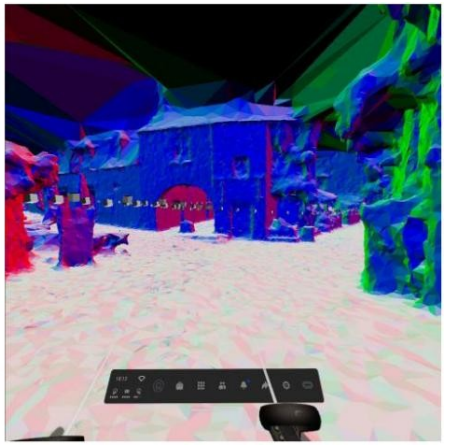
They are generated for several VR headsets thanks to Unity

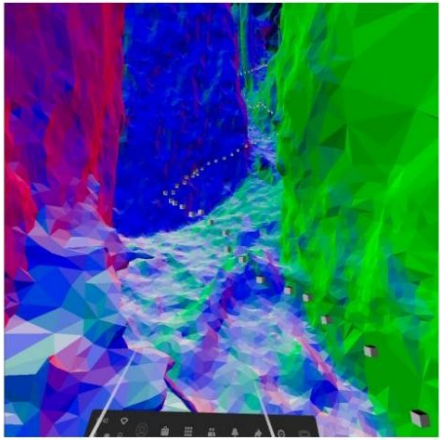
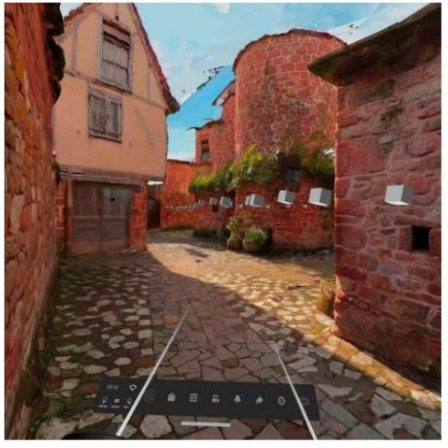
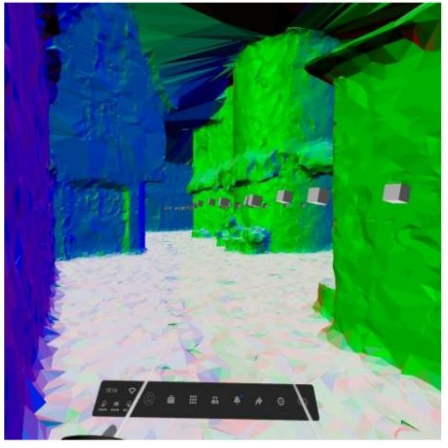
(Available today: standalone Oculus Quest and Go)

(Available in the next few days: lot’s of VR headsets linked to a PC)

VR applications (screenshots of Oculus Quest)



Conclusion

The paper estimates the vertical direction and the scale of a triangulated surface reconstructed by photogrammetry, using 3D PCA and 2D Hough

This allows to reset the coordinate system for VR applications of immersive visualisation, which are available on the web

The paper experiments on non-trivial environments: a basalt canyon and a medieval city, whose acquisition trajectories are 800m and 1.6km long

Limitation: the vertical cannot be computed if all vertical triangles have the same direction (eg a set of parallel walls and facades)

Improvements include

- comparisons to results obtained by other sensors (GPS, IMU)
- decrease in cybersickness without sacrifying the spatial awareness