Proxy Clouds for RGB-D Stream Processing: A Preview

Adrien Kaiser[1,2] Jose Alonso Ybanez Zepeda[2] Tamy Boubekeur[1]

[1]: LTCI, Telecom ParisTech, Paris-Saclay University
[2]: Ayotle SAS

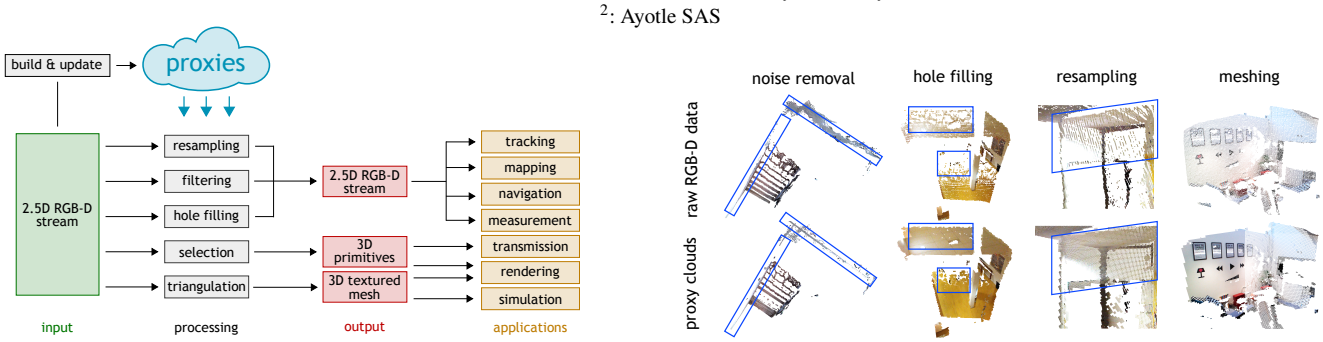

Figure 1: ***(Left) Proxy Clouds Workflow.*** *From a stream of RGB-D frames, proxies are built and updated through time (Sec 2.1). They are used as priors to process the frames (resampling, filtering or hole filling) for better tracking, mapping, automated navigation or measurement. A selection of proxies based on the current RGB-D frame can be used for lightening data transmission or as triangulation prior for fast depth data meshing, with application to rendering or simulation.* ***(Right) Data Improvement.*** *Raw RGB-D data (top) and* Proxy Clouds*-improved data after 100 frames (bottom) showing results of real time noise removal, hole filling, point cloud resampling and meshing. Blue surrounded areas highlight regions where improvement using* Proxy Clouds *is significant compared to the low quality input RGB-D frames.*

Abstract

Modern consumer depth cameras are widely used for 3D capture in indoor environments, for applications such as modeling, robotics or gaming. Nevertheless, their use is limited by their low resolution, with frames often corrupted with noise, missing data and temporal inconsistencies. In order to cope with all these issues, we present Proxy Clouds*, a multiplanar superstructure for real-time processing of RGB-D data. By generating a single set of planar proxies from raw RGB-D data and updating it through time, several processing primitives can be applied to improve the quality of the RGB-D stream or lighten further operations. We illustrate the use of* Proxy Clouds *on several applications, including noise and temporal flickering removal, hole filling, resampling, color processing and compression. We present experiments performed with our framework in indoor scenes of different natures captured with a consumer depth sensor.*

Categories and Subject Descriptors (according to ACM CCS): I.4.3 [Computing Methodologies / Image Processing and Computer Vision]: Enhancement—Geometric Correction

1. Introduction

1.1. Objectives

Modern consumer depth cameras are attractive with an affordable price and many possible applications of their real time RGB-D stream output, ranging from human computer interaction to augmented reality, through geometry capture. Although such technologies made great progress over the last decade, the limited quality of their RGB-D stream still limit their application spectrum. It mostly originates in the low resolution of the frames and the inherent noise, incompleteness and temporal inconsistency attached to single view capture.

Proxy Clouds aim at analyzing and structuring such streams to improve them on the fly, under real time embedded constraints with limited memory. They take the form of a lightweight planar superstructure stable through time which gives priors to apply several processing primitives to the RGB-D frames (Sec. 2.2), to reinforce the data and simplify or lighten subsequent operations. Our system takes a raw RGB-D stream as input and outputs an enhanced RGB-D stream together with the optional set of proxies associated with the current RGB-D data (see the workflow in Fig. 1 left).

1.2. Previous Work

Plane Detection in RGB-D Stream Methods that build high level models of captured 3D data are mostly based on *RANSAC*, the *Hough transform* or *Region Growing* algorithms. In our embedded, real time, memory-limited context, we take inspiration from the RANSAC-based method proposed by Schnabel et al. [SWK07] for its time and memory efficiency, by repeating plane detection through time to acquire a consistent model and cope with the stochastic nature of RANSAC.

DOI: 10.2312/egp.20171039

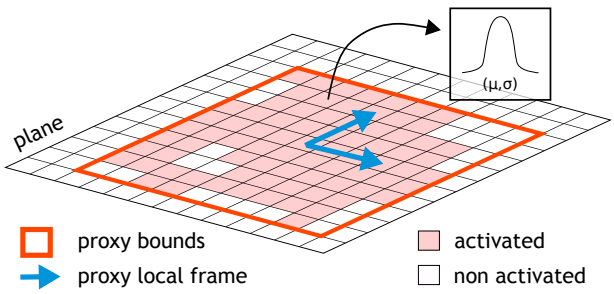

Figure 2: ***Planar Proxy Model.*** *Built upon a plane in 3D space, the model is made of a local frame, bounds and a grid of cells which contain statistics extracted from the RGB-D data. Activated cells are the ones containing inliers from many frames.*

RGB-D Stream Processing Depth maps can be denoised using spatial filters e.g., gaussian, median, bilateral, adaptive or anisotropic filters, often refined through time. Other structural methods include over-segmentation and region-growing. They can be upsampled using *joint bilateral upsampling* or *weighted mode filtering*. Their holes can be filled with either spatial or morphological filters, together with inpainting or multiscale processing for e.g., *depth image-based rendering* (DIBR) under close viewing conditions. Only a few methods have used planar proxies as prior to process 3D data, with in particular Schnabel et al. [SDK09] who detect limits of planes to fill in holes in 3D point clouds.

2. Proxy Clouds

2.1. Model

Basically, *Proxy Clouds* model RGB-D data which is often seen and consistent through frames, hence capture the dominant structural elements of the scene. To do so, they take the form of a multiplanar superstructure, where each planar proxy is equipped with a local frame, bounds and, within the bounds, a regular 2D grid of rich statistics extracted from the RGB-D data, including mean and variance for color and depth (see Fig. 2).

We build planar proxies and update them through time using solely incoming raw RGB-D frames from the live stream. More precisely, for each new RGB-D image $X_t = \{I_t, D_t\}$ (color and depth), we run the following algorithm:

1. Filter D_t through Gaussian image and depth bilateral filtering
2. Estimate the normal field N_t from D_t using the depth gradient
3. Estimate the camera motion M_t from the previous frame using point features from I_t [EHS*14]
4. Search X_t for previous proxies
   - 4.1. Register previous frame proxies to X_t using M_t
   - 4.2. Cast a vote from samples of X_t to the proxy they are inlier of
   - 4.3. Update with X_t – or discard proxies given their vote count
5. Detect new proxy planes in $X_t \setminus$ *inliers* using RANSAC [SWK07] and initialize local frames and statistics with X_t
6. Refine camera motion M_t using proxies

2.2. RGB-D Stream Processing

Our proxy cloud allows recovering the underlying structure of planar data and is used as a prior to apply different types of processing to incoming RGB-D frames. First, projecting data points onto a proxy or using it as a high level range space for cross bilateral filtering allows **removing the acquisition noise** due to the sensor. Second, the proxies consolidate the geometry of the scene by accumulating observations in multiple frames, which makes them stable with **no temporal flickering**. Third, the stable proxies define a support updated at each frame to **reinforce missing data and fill holes**. Forth, RGB-D streams can be **super-sampled** on the fly, by enriching the low definition geometric component using the higher resolution color component structured in the proxies. Fifth, proxies provide a prior to **apply color processing** to the RGB-D stream and define a support upon which known image operations such as blurring or sharpening can be performed. Last, the proxy cloud is a **compressed lightweight geometric substitute** to the huge amount of depth data carried in the RGB-D stream and avoids storing uncertain depth components such as highly noisy depth regions. The data can then be bilateraly upsampled back to high resolution. Figure 1 (right) shows examples of live data improvement.

3. Results and Discussion

Proxy Clouds are currently implemented through hardware and software components. The hardware setup is made of an Orbbec Astra † RGB-D sensor and a mobile device with Intel Core i7, 8 cores at 2.0GHz and 4GB memory. The software setup has a client-server architecture, where the server runs on an embedded environment to trigger the sensor and process the data. The client is a GUI that allows controlling the processing parameters and getting a real time feedback of the stream. A limited range of intuitive parameters allow the user to control the trade-off between quality of the output and performance of the processing. The current timing to build and update planar proxies using our implementation is around 200 ms for an input depth image of 320x240 pixels.

Future works are oriented towards performance in order to bring the stream processing to a high rate on embedded platforms. For now, the proxy grid is regular and statistics are stored for each cell. One of the next steps would be to ligthen this 2D representation by using e.g. sparse surfel quadtrees. Also, the use of more complex geometric primitives such as cylinders, spheres, cones or ellipsoids would allow modeling more objects seen in indoor scenes.

References

[EHS*14] ENDRES F., HESS J., STURM J., CREMERS D., BURGARD W.: 3-d mapping with an rgb-d camera. *IEEE Transactions on Robotics 30*, 1 (2014), 177–187. 2

[SDK09] SCHNABEL R., DEGENER P., KLEIN R.: Completion and reconstruction with primitive shapes. *Computer Graphics Forum 28*, 2 (2009), 503–512. 2

[SWK07] SCHNABEL R., WAHL R., KLEIN R.: Efficient ransac for point-cloud shape detection. *Computer Graphics Forum 26*, 2 (June 2007), 214–226. 1, 2

† Orbbec Astra: https://orbbec3d.com/product-astra/