Volume 0 (1981), Number 0 pp. 1–5 COMPUTER GRAPHICS *forum*

Detection of geometric temporal changes in point clouds - Additional Material

Gianpaolo Palma[1], Paolo Cignoni[1], Tamy Boubekeur[2], Roberto Scopigno[1]

[1]Visual Computing Lab - ISTI - CNR, Italy
[2]Telecom ParisTech - CNRS LTCI - Institut Mines Telecom, France

Abstract
This document contains additional results of the paper "Detection of geometric temporal changes in point clouds"

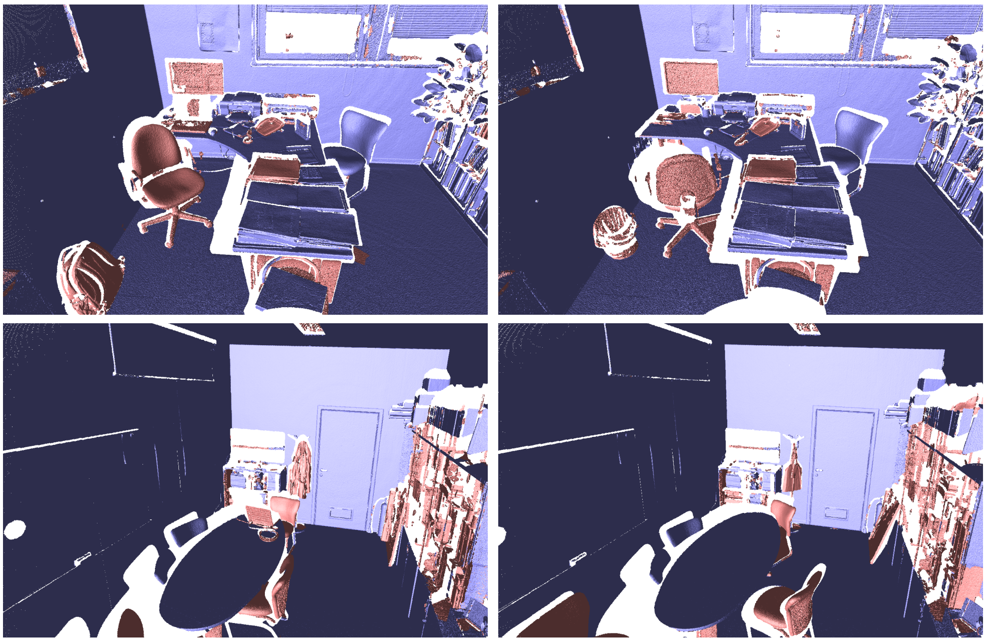

Figure 1: *Change detection results between OFFICE T0 (Left) and OFFICE T1 (Right) in two different places. More info in the 1st row of the Table 1.*

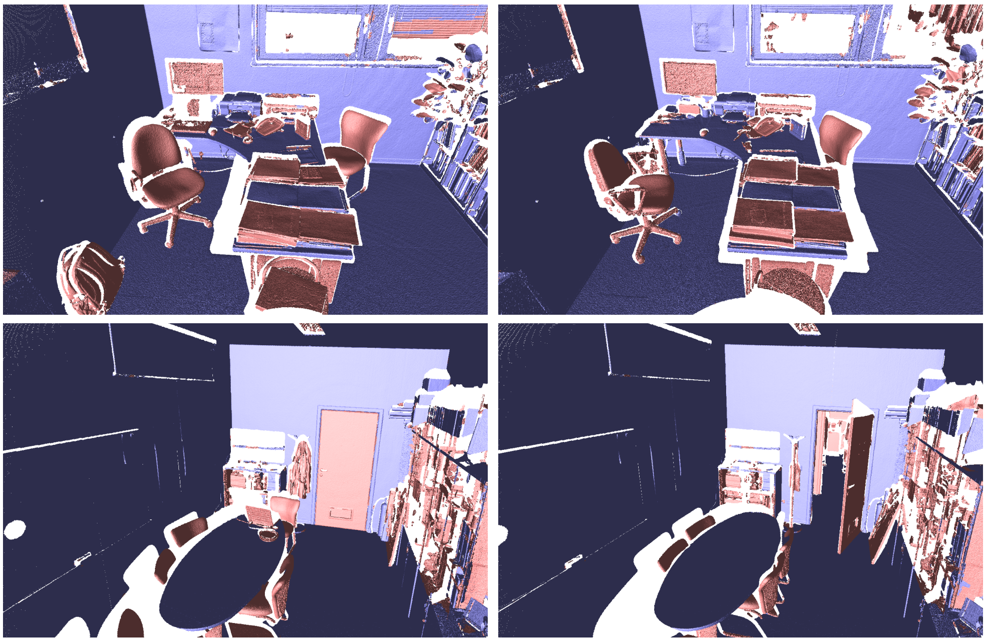

Figure 2: *Change detection results between OFFICE T0 (Left) and OFFICE T2 (Right) in two different places. More info in the 2nd row of the Table 1.*

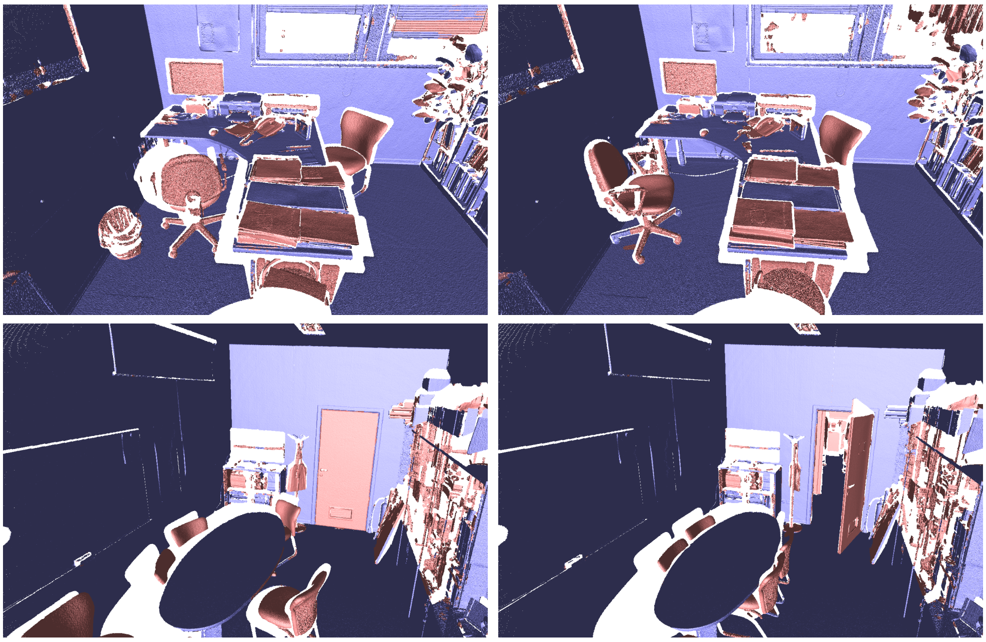

Figure 3: *Change detection results between OFFICE T1 (Left) and OFFICE T2 (Right) in two different places. More info in the 3rd row of the Table 1.*

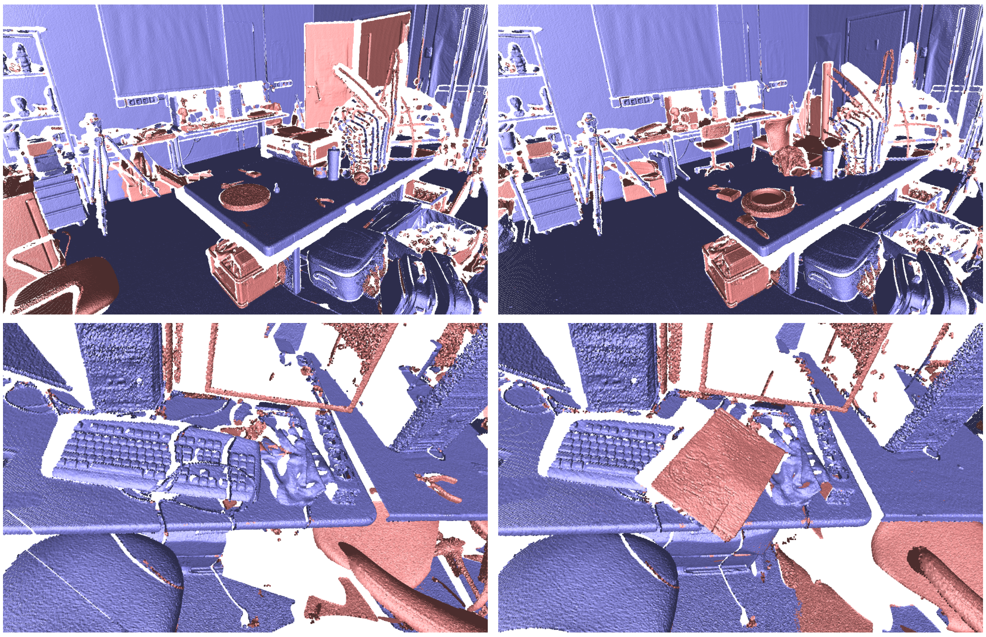

Figure 4: *Change detection results between LAB T0 (Left) and LAB T1 (Right) in two different places. More info in the 2nd row of the Table 1.*

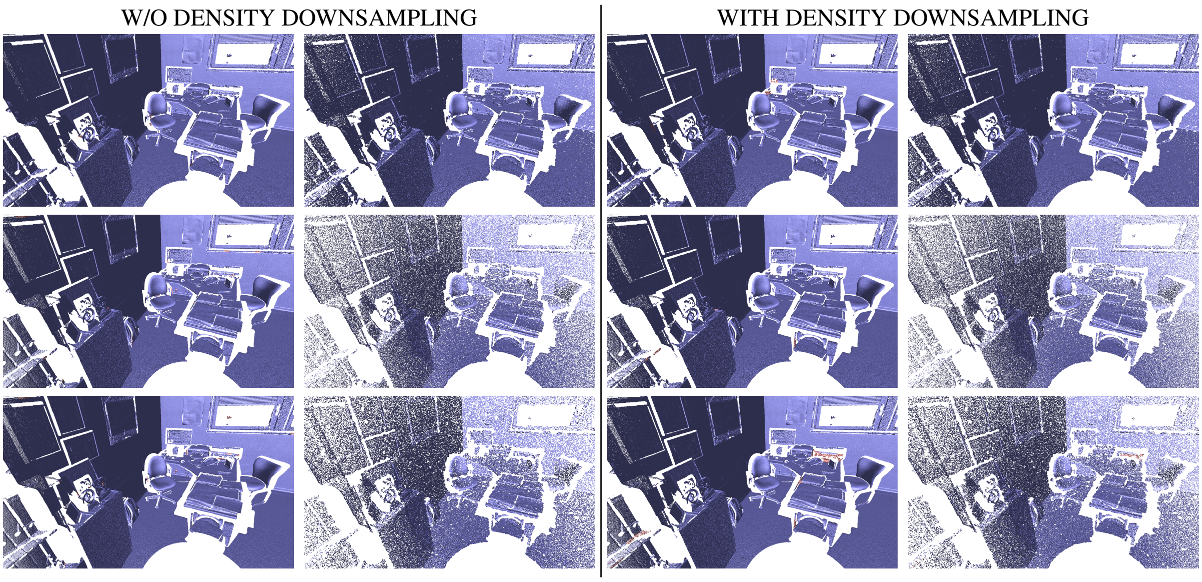

Figure 5: *Change detection between a scan and its sub-sampled versions to verify the robustness against the point density. (First row) OFFICE S1 vs OFFICE S0.5. (Second row) OFFICE S1 vs OFFICE S0.25. (Third row) OFFICE S1 vs OFFICE S0.125. The left column shows the result obtained using directly the two scans while the right column shows the result with the sub-sampling preprocessing to reduce the scans at the same local density.*

	Cloud1	Cloud2	Outliers1	Outliers2	Changes1	Changes2	Time
OFFICE T0 - OFFICE T1	6354k	6363k	293k (4.61%)	292k (4.58%)	517k (8.13%)	470k (7.38%)	251 sec
OFFICE T0 - OFFICE T2	6354k	6358k	293k (4.61%)	299k (4.70%)	732k (11.58%)	640k (10.06%)	339 sec
OFFICE T1- OFFICE T2	6363k	6358k	292k (4.58%)	299k (4.70%)	607k (9.53%)	566k(8.9%)	297 sec
LAB T0 - LAB T1	6226k	6194k	328k (5.26%)	327k (5.27%)	855k (13.73%)	695k (11.22%)	368 sec
OFFICE S1 - OFFICE S0.5	6354k	3177k	293k (4.61%)	116k (3.65%)	1665 (0.0026%)	0 (0%)	140 sec
OFFICE S1 - OFFICE S0.25	6354k	1588k	293k (4.61%)	57782 (3.63%)	12945 (0.002%)	650 (0.0038%)	105 sec
OFFICE S1 - OFFICE S0.125	6354k	794k	293k (4.61%)	30085 (3.70%)	10525 (0.15%)	897 (0.11%)	88 sec
OFFICE S1 - OFFICE S0.5 DOWN	6354k	3177k	293k (4.61%)	116k (3.65%)	6259 (0.098%)	3196 (0.1%)	99 sec
OFFICE S1 - OFFICE S0.25 DOWN	6354k	1588k	293k (4.61%)	57782 (3.65%)	15310 (0.24%)	3782 (0.23%)	64 sec
OFFICE S1 - OFFICE S0.125 DOWN	6354k	794k	293k (4.61%)	30085 (3.70%)	36989 (0.56%)	5217 (0.62%)	55 sec
OFFICE P1 - OFFICE P2	6280k	6283k	300k (4.77%)	303k (4.82%)	1084k (17.26%)	984k (15.66%)	504 sec
OFFICE P1 - OFFICE P2 DOWN	6280k	6283k	300k (4.77%)	303k (4.82%)	1100k (17.51%)	820k (13.05%)	112 sec

Table 1: *Test case and performance data*

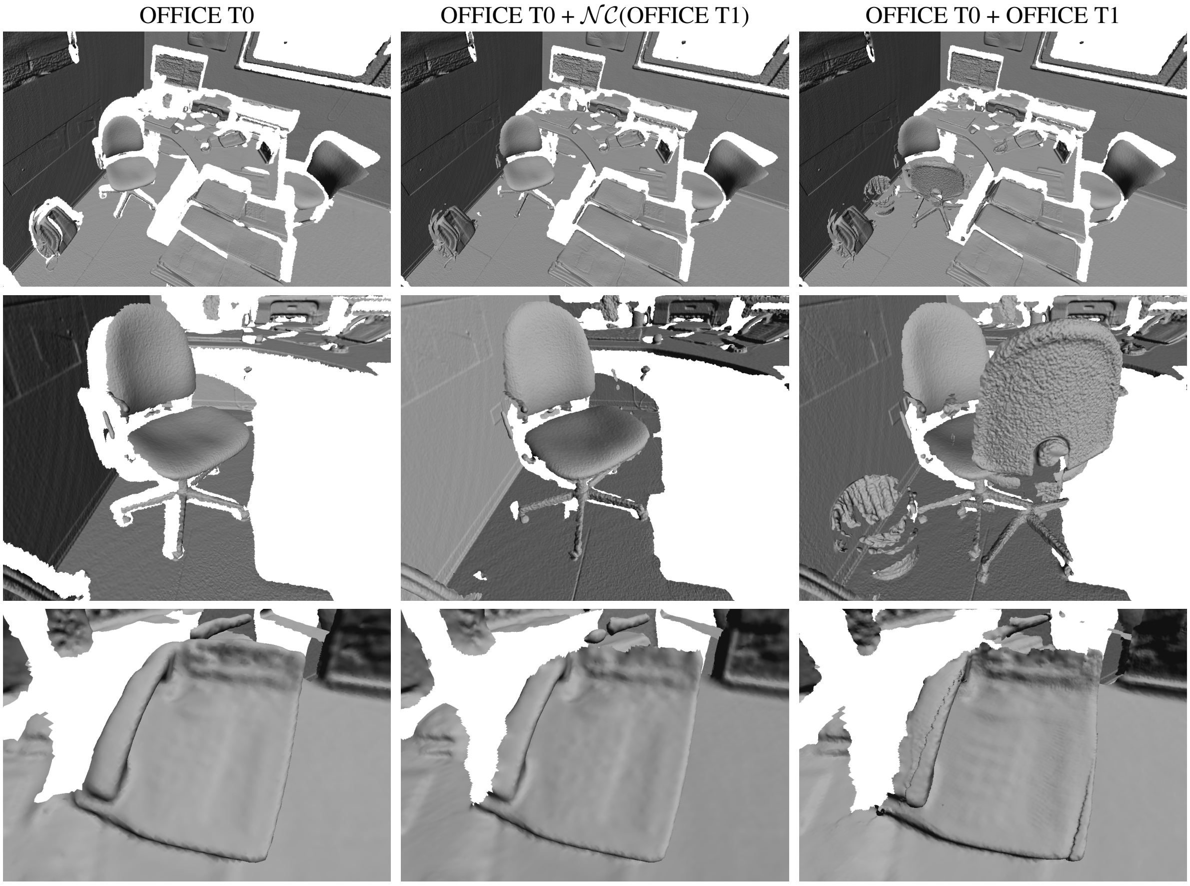

Figure 6: *Triangulation results from three inputs: (Left) OFFICE T0; (Center) OFFICE T0 plus the no change regions of OFFICE T1; (Right) union of OFFICE T0 and OFFICE T1. The pictures in the center show a more complete reconstruction without inconsistencies due to the intersection of different objects.*

Appendix A

This appendix contains the pseudo-code and notation of the algorithm described in Section 6 of the paper. The function FITALGEBRSPHERE(Q) uses the method described in [GGG08] to fit an algebraic sphere using the point in the set Q.

References

[GGG08] GUENNEBAUD G., GERMANN M., GROSS M. H.: Dynamic sampling and rendering of algebraic point set surfaces. *Compututer Graphics Forum 27*, 2 (2008), 653–662. 5

A, B	point clouds
$\mathcal{C}(A)$	subset of A classified as change
$\mathcal{NC}(A)$	subset of A classified as no-change
$\mathbf{x}$	point of the cloud
$\vec{n}_\mathbf{x}$	normal of the point $\mathbf{x}$
s_u	algebraic sphere
$s_\mathbf{q}^A$	algebraic sphere computed around $\mathbf{q}$ using the point in the cloud A
$s_\mathbf{q}^A[i]$	i-th algebraic sphere of a GLS descriptor computed around $\mathbf{q}$ using the point in the cloud A
$radius[\mathbf{x}]$	radius of the point $\mathbf{x}$
$dist[\mathbf{x}]$	distance of the point $\mathbf{x}$ from the nearest no-change point in the other cloud according the Geodesic distance
$nearGeo[\mathbf{x}]$	pointer to the the nearest no-change point in the other cloud according the Geodesic distance
$indeg[\mathbf{x}]$	in-degree of the point $\mathbf{x}$ in the temporal proximity graph
$updateToC[\mathbf{x}]$	it is true if $\mathbf{x}$ must modify its state from no-change to change
$updateToNC[\mathbf{x}]$	it is true if $\mathbf{x}$ must modify its state from change to no-change

Table 2: *Notation*

Algorithm 1 Build the temporal proximity graph

function BUILDPROXIMITYGRAPH(A, B)
    for all $\mathbf{x} \in \mathcal{C}(A)$ **do**
        $N \leftarrow \{\mathbf{y} \in \mathcal{NC}(B) \mid \|\mathbf{y}-\mathbf{x}\| \leq radius[\mathbf{x}]\}$
        if $N \neq \emptyset$ **then**
            $dist[\mathbf{x}] \leftarrow \min_{\mathbf{y} \in N} \|\mathbf{y}-\mathbf{x}\|$
            $nearGeo[\mathbf{x}] \leftarrow \arg\min_{\mathbf{y} \in N} \|\mathbf{y}-\mathbf{x}\|$
        else
            $dist[\mathbf{x}] \leftarrow \infty$
            $nearGeo[\mathbf{x}] \leftarrow$ null
    $update \leftarrow$ true
    while $update$ **do**
        $update \leftarrow$ false
        for all $\mathbf{x} \in \mathcal{C}(A)$ **do**
            $N \leftarrow \{\mathbf{y} \in \mathcal{C}(A) \mid \|\mathbf{y}-\mathbf{x}\| \leq radius[\mathbf{x}]\}$
            for all $\mathbf{y} \in N$ **do**
                if $dist[\mathbf{y}] + \|\mathbf{y}-\mathbf{x}\| < dist[\mathbf{x}]$ **then**
                    $dist[\mathbf{x}] \leftarrow dist[\mathbf{y}] + \|\mathbf{y}-\mathbf{x}\|$
                    $nearGeo[\mathbf{x}] \leftarrow nearGeo[\mathbf{y}]$
                    $update \leftarrow$ true

Algorithm 2 Check if the point $\mathbf{y}$ is near to the plane defined by $\mathbf{x}$

function CHECKPLANE($\mathbf{x}, \mathbf{y}$)
    return $\vec{n}_\mathbf{x} \cdot \vec{n}_\mathbf{y} > t_1 \wedge |(\mathbf{y}-\mathbf{x}) \cdot \vec{n}_\mathbf{x}| < t_2$

Algorithm 3 Check if the point $\mathbf{x}$ is a double time accumulation vertex in the proximity graph

function $\textsc{IsDoubleAccumP}(\mathbf{x}, A)$
  if $indeg[\mathbf{x}] > 1$ **then**
    $N_x \leftarrow \{\mathbf{y} \in A \mid \|\mathbf{y}-\mathbf{x}\| \leq radius[\mathbf{x}] \wedge indeg[\mathbf{y}] > 1\}$
    if $N_x \neq \emptyset$ **then**
      return true
  return false

Algorithm 4 Propagate the no-change information

function $\textsc{PropagateNoChange}(A)$
  for all $\mathbf{x} \in A$ **do**
    $updateToNC[\mathbf{x}] \leftarrow$ false
  for all $\mathbf{x} \in \mathcal{C}(A)$ **do**
    if $\neg\textsc{IsDoubleAccumP}(nearGeo[\mathbf{x}], A)$ **then**
      $updateToNC[\mathbf{x}] \leftarrow$ true
  for all $\mathbf{x} \in \mathcal{NC}(A)$ **do**
    if $\neg\, updateToC[\mathbf{x}]$ **then**
      $updateToNC[\mathbf{x}] \leftarrow$ true
  $update \leftarrow$ true
  while $update$ **do**
    $update \leftarrow$ false
    for all $\mathbf{x} \in A$ **do**
      if $updateToNC[\mathbf{x}]$ **then**
        $N \leftarrow \{\mathbf{p} \in \mathcal{C}(A) \mid \|\mathbf{p}-\mathbf{x}\| \leq radius[\mathbf{x}]\}$
        for all $\mathbf{p} \in N$ **do**
          if $\textsc{CheckPlane}(\mathbf{x}, \mathbf{p})$ **then**
            $updateToNC[\mathbf{p}] \leftarrow$ true
            $update \leftarrow$ true

Algorithm 5 Propagate the change information

function $\textsc{PropagateChange}(A, B)$
  for all $\mathbf{x} \in A$ **do**
    $updateToC[\mathbf{x}] \leftarrow$ false
  for all $\mathbf{x} \in \mathcal{C}(A)$ **do**
    if $\textsc{IsDoubleAccumP}(nearGeo[\mathbf{x}], A)$ **then**
      $\mathbf{y} \leftarrow \arg\min_{\mathbf{y} \in B} \|\mathbf{y}-\mathbf{x}\|$
      $Q \leftarrow \{\mathbf{p} \in B \mid \|\mathbf{p}-\mathbf{y}\| \leq \|\mathbf{y}-\mathbf{x}\|\}$
      $s_u \leftarrow [u_c\ \mathbf{u}_l\ u_q] \leftarrow \textsc{FitAlgebrSphere}(Q)$
      if $u_q s_u(\mathbf{x}) > 0$ **then** $\triangleright$ outside the volume?
        $count \leftarrow \textsc{Propagate}(\mathbf{x}, A, B)$
  $update \leftarrow$ true
  while $update$ **do**
    $update \leftarrow$ false
    for all $\mathbf{x} \in A$ **do**
      if $updateToC[\mathbf{x}]$ **then**
        $temp \leftarrow \textsc{Propagate}(\mathbf{x}, A, B)$
        $update \leftarrow update \vee (temp > 0)$
        $count \leftarrow count + temp$
  return $count$

function $\textsc{Propagate}(\mathbf{x}, A, B)$
  $count \leftarrow 0$
  $N \leftarrow \{\mathbf{p} \in \mathcal{NC}(A) \mid \|\mathbf{p}-\mathbf{x}\| \leq radius[\mathbf{x}]\}$
  for all $\mathbf{p} \in N$ **do**
    if $\textsc{CheckPlane}(\mathbf{x}, \mathbf{p})$ **then**
      $updateToC[\mathbf{p}] \leftarrow$ true
      $count \leftarrow count + 1$
      $\mathbf{b} \leftarrow \arg\min_{\mathbf{y} \in B} \|\mathbf{y}-\mathbf{p}\|$
      if $\textsc{CheckPlane}(\mathbf{p}, \mathbf{b})$ **then**
        $updateToC[\mathbf{b}] \leftarrow$ true
  return $count$