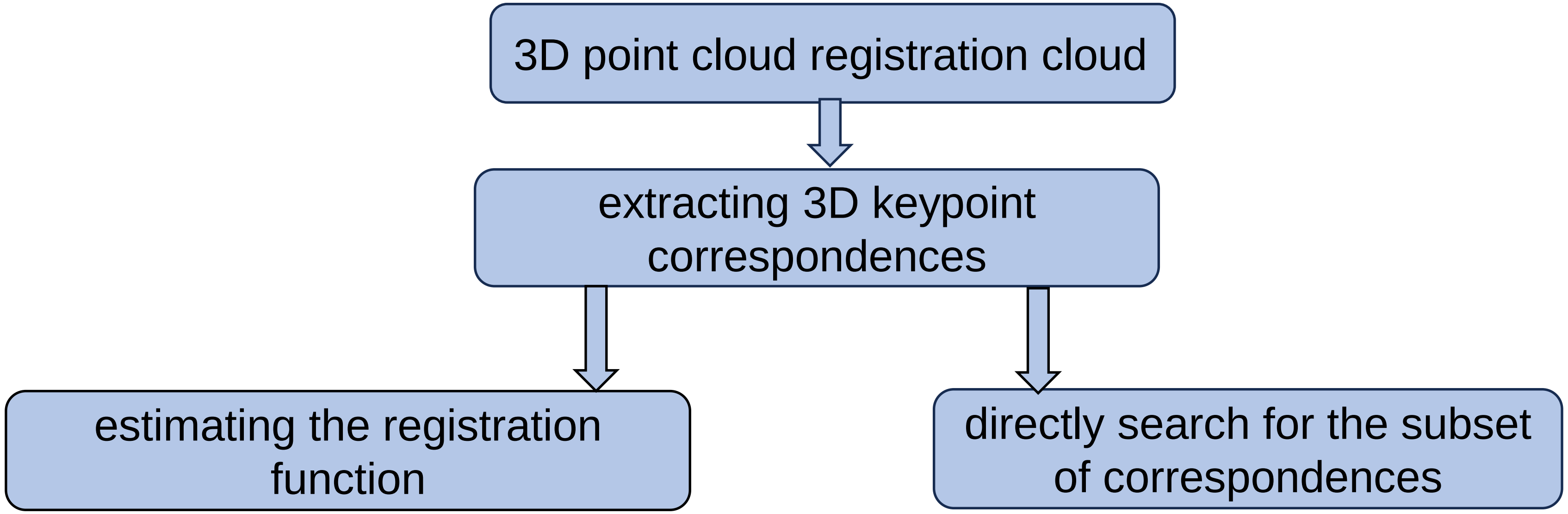
Mutual Voting for Ranking 3D Correspondences

翁霄羽
wengxiaoyu2009@163.com

INTRODUCTION

A Practical Maximum Clique Algorithm for Matching with Pairwise Constraints :A way for 3D point cloud registration



keypoint-based 3D registration

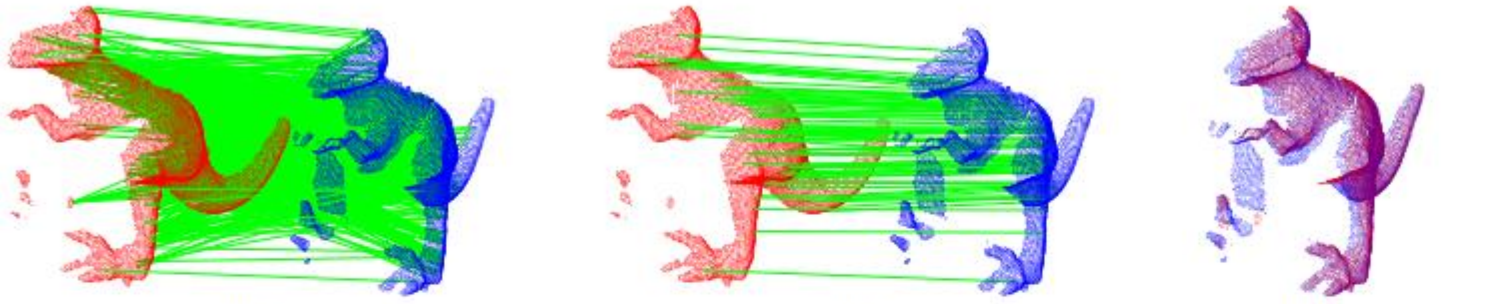
A popular paradigm:

1. let $X=\{x_i\}_{i=1}^{n}$ and $Y=\{y_j\}_{j=1}^{n}$ be two input point clouds
2. generate a tentative correspondence set $C=\{c_k\}_{k=1}^{N}$
3. $C_k=(x_k, y_{k'})$ associates a point $x_k \in X, y_{k'} \in Y$
4. a 6 DoF rigid transformation can be estimated from C

Disadvantage: outliers exist in C, thus the registration function must be estimated using a robust technique.

It is vital to investigate keypoint-based 3D registration

Find the largest subset of C that are pairwise consistent



(a) (b) (c)

Figure 1. Example registration. (a) Input correspondence set $\mathcal{C}$ of size $N = 2000$. (b) The largest subset of $\mathcal{C}$ that are pairwise consistent (found in 0.06 seconds by our novel max clique algorithm). (c) The alignment estimated via SVD [7] using the correspondences in (b).

Solve:

$$\begin{aligned} \underset{\mathcal{I} \subseteq \{1,\ldots,N\}}{\text{maximise}} \quad & |\mathcal{I}| \\ \text{subject to} \quad & d(c_i, c_j) \le \epsilon, \ \forall i, j \in \mathcal{I}, \end{aligned} \tag{1}$$

where the "distance" between two correspondences $c_i = (\mathbf{x}_i, \mathbf{y}_{i'})$ and $c_j = (\mathbf{x}_j, \mathbf{y}_{j'})$ is given by

$$d(c_i, c_j) = \left| \|\mathbf{x}_i - \mathbf{x}_j\|_2 - \|\mathbf{y}_{i'} - \mathbf{y}_{j'}\|_2 \right|. \tag{2}$$

1.the registration function can be estimated from the data indexed by $I*$
2. value $I*$ can directly be taken as the similarity score of shapes X and Y

um 澳大

Graph formulation

Let G = (V , E) represent an undirected graph with vertices V = {vi} and edges E = {(vi,vj)}.

Definition 2.1 (Adjacency and degree). We say that a pair of vertices v_i and v_j of G are adjacent if $(v_i, v_j) \in E$. For each $v_i \in V$, denote the adjacency of v_i as

$$\Gamma(v_i) = \{v_j \in V \mid (v_i, v_j) \in E\}. \tag{3}$$

Then $|\Gamma(v_i)|$ is called the degree of v_i.

Definition 2.2 (Consistency graph). Given a set of correspondences $\mathcal{C}$, the consistency graph is constructed as the graph with vertices $V = \mathcal{C}$ and edges

$$E = \{(c_i, c_j) \in \mathcal{C} \times \mathcal{C} \mid d(c_i, c_j) \leq \epsilon,\ i \neq j\}, \tag{4}$$

i.e., two correspondences c_i and c_j are adjacent in the graph if they are pairwise consistent.

Definition 2.3 (Inconsistency graph). The inconsistency graph is the complement of the consistency graph, i.e., the graph with vertices $V = \mathcal{C}$ and edges

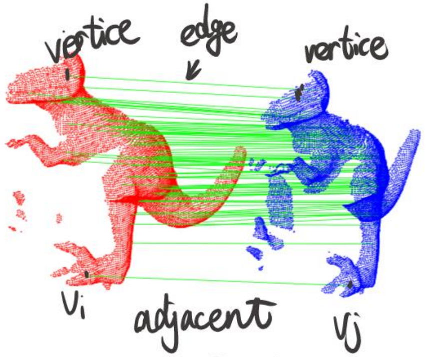
$$E = \{(c_i, c_j) \in \mathcal{C} \times \mathcal{C} \mid d(c_i, c_j) > \epsilon,\ i \neq j\}. \tag{5}$$

i.e., two correspondences c_i and c_j are adjacent in the graph if they are pairwise inconsistent.

Definition 2.4 (Clique). A clique of a graph $G = (V, E)$ is a subgraph of G where every pair of vertices in the subgraph are adjacent. A *maximum clique (MC)* of G is a clique of G with the largest size.

Definition 2.5 (Vertex cover). A vertex cover of a graph $G = (V, E)$ is a subset of V such that every edge in E is incident with at least one vertex in the subset. The removal of a vertex cover from G leaves an independent set, i.e., a set of vertices with no edges. A *minimum vertex cover (MVC)* of G is a vertex cover of G with the smallest size.

eg: consistency graph:



Vi, Vj 同属一条边.

$$\begin{aligned} \underset{\mathcal{I} \subseteq \{1,\ldots,N\}}{\text{maximise}} \quad & |\mathcal{I}| \\ \text{subject to} \quad & d(c_i, c_j) \leq \epsilon,\ \forall i, j \in \mathcal{I}, \end{aligned}$$

finding the MC of the *consistency* graph constructed from the correspondence set *C*

MIP solutions

MC can be written as the MIP

$$\begin{aligned} &\text{maximise} && \sum_{i=1}^{|V|} x_i \\ &\text{subject to} && x_i + x_j \leq 1,\ \forall (v_i, v_j) \notin E \\ & && x_i \in \{0,1\},\ i = 1 \ldots |V|. \end{aligned}$$

当vi、vj不属于一条边时，xi、xj不同时为1，所有的x_i=1组成MC

The MIP formulation for MVC is

$$\begin{aligned} &\text{minimise} && \sum_{i=1}^{|V|} x_i \\ &\text{subject to} && x_i + x_j \geq 1,\ \forall (v_i, v_j) \in E \\ & && x_i \in \{0,1\},\ i = 1 \ldots |V|. \end{aligned}$$

BNB

BnB explores the set of cliques of *G* by building a search tree over the vertices *V*

A fundamental aspect of BnB algorithms is to prunebranches in the search tree that are not promising.

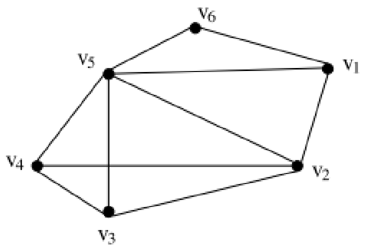


Figure 2. A sample input graph $G = (V, E)$.

Algorithm 1 Basic BnB algorithm for MC.

Require: A set of candidate vertices S.

global variables: The current clique R and the best clique found so far R_{best}.
initialisation: $R \leftarrow \emptyset$, $R_{best} \leftarrow \emptyset$.
while $S \neq \emptyset$ **do**
    if $|R| + |S| \leq |R_{best}|$ **then**
        return
    end if
    $v \leftarrow$ first vertex in S.
    $R \leftarrow R \cup \{v\}$.
    $S' \leftarrow S \cap \Gamma(v)$.
    if $S' \neq \emptyset$ **then**
        Recursive call with candidate vertices S'.
    else if $|R| > |R_{best}|$ **then**
        $R_{best} \leftarrow R$.
    end if
    $R \leftarrow R \setminus \{v\}$.
    $S \leftarrow S \setminus \{v\}$.
end while
return R_{best}.

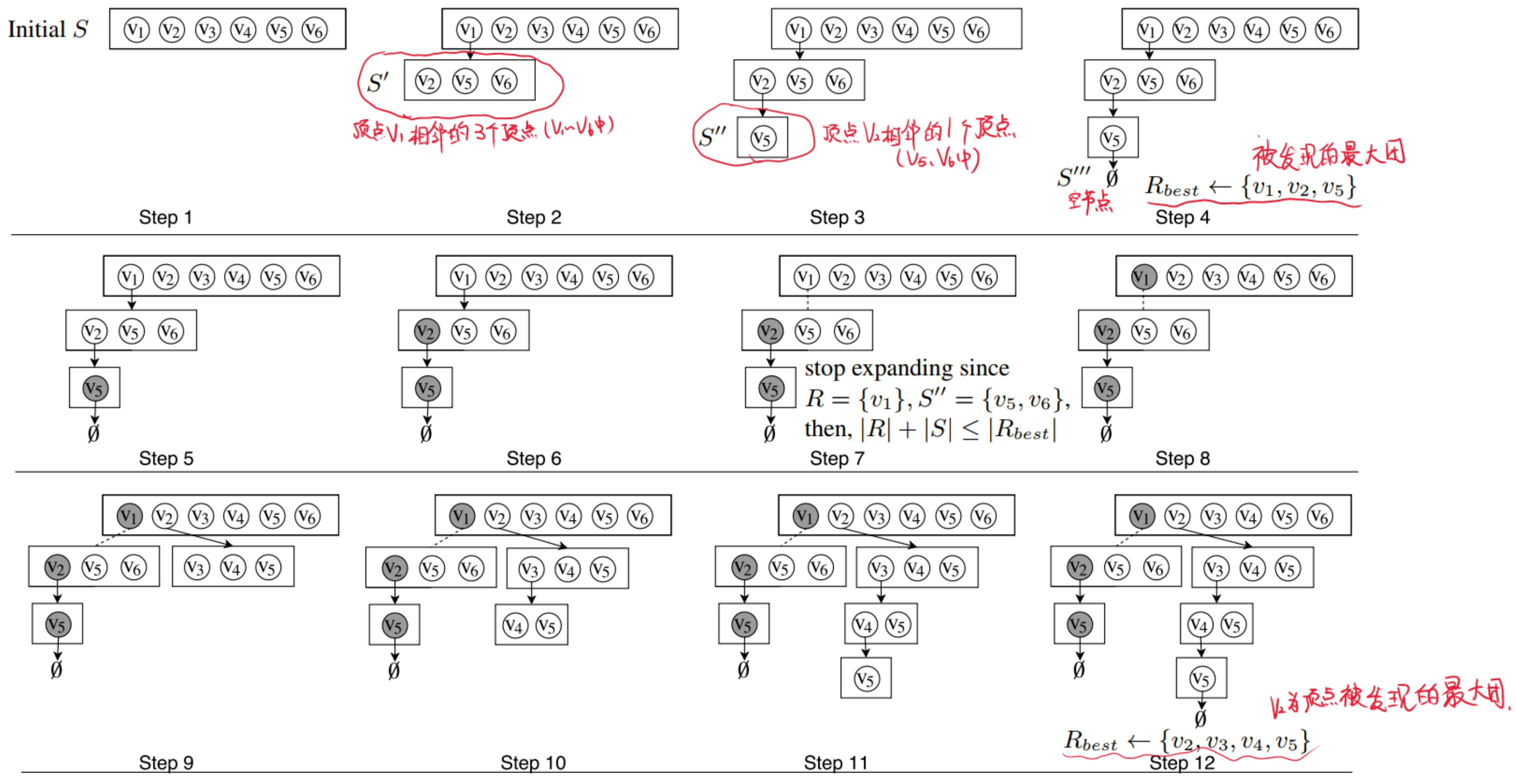


Figure 3. 在求解图2中的示例图时，算法1的搜索树的进展。前12步是逐步生成的，以显示顶点扩展、顶点删除(灰色顶点)和停止扩展节点(虚线)。树的叶结点对应于一个空集合。

MCQ

A colouring of a graph *G* = (*V; E*) is a labelling *f* of itsvertices such that no adjacent vertices have the same colour $f(v_i) \neq f(v_j)$ for all $(v_i, v_j) \in E$,

Algorithm 2 MCQ algorithm for MC.

Require: A set of candidate vertices S, a colouring f.

global variables: The current clique R and the best clique found so far R_{best}.
initialisation: $R \leftarrow \emptyset$, $R_{best} \leftarrow \emptyset$.
initialisation: Reorder vertices in S in descending order of degree, i.e., $|\Gamma(v_i)| \geq |\Gamma(v_j)|$, $\forall v_i, v_i \in S$ if $i < j$.
while $S \neq \emptyset$ **do**
    $v \leftarrow$ last vertex in S.
    if $|R| + f(v) \leq |R_{best}|$ **then**
        return
    end if
    $R \leftarrow R \cup \{v\}$.
    $S' \leftarrow S \cap \Gamma(v)$.
    if $S' \neq \emptyset$ **then**
        Find a colouring f' of S'.
        Recursive call with candidate vertices S' and colouring f'.
    else if $|R| > |R_{best}|$ **then**
        $R_{best} \leftarrow R$.
    end if
    $R \leftarrow R \setminus \{v\}$.
    $S \leftarrow S \setminus \{v\}$.
end while
return R_{best}.

find the colouring with the minimum number of colours

$$\min_f \max_{v_i \in S} f(v_i).$$

PMC algorithm for MC

To speed up MCQ, we introduce a novel extra pruning step to avoid exploring mul

branches during the search for the optimal solution

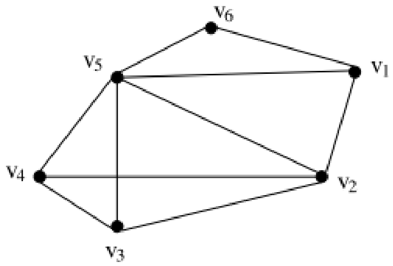


Figure 2. A sample input graph $G = (V, E)$.

Algorithm 3 PMC algorithm for MC.

Require: A set of candidate vertices S, removed vertices F, a colouring f.

global variables: The current clique R and the best clique found so far R_{best}

initialisation: $R \leftarrow \emptyset$, $R_{best} \leftarrow \emptyset$.

initialisation: Reorder vertices in S (e.g., as in Line 3 of Algorithm 2).

Find $C \subseteq S$ as described in Sec. 4.

$i \leftarrow |S|$.

while $i > 0$ **do**

  if $|R| + \max_{v_j \in S} f(v_j) \le |R_{best}|$ **then**

    return

  end if

  $v_i \leftarrow$ i-th vertex of S.

  if $v_i \in C$ **then**

    $R \leftarrow R \cup \{v_i\}$.

    $S' \leftarrow S \cap \Gamma(v_i)$.

    if $S' \neq \emptyset$ **then**

      $F' \leftarrow F \cap \Gamma(v_i)$.

      Find a colouring f' of S'.

      Recursive call with candidate vertices S', removed vertices F' and colouring f'.

    else if $|R| > |R_{best}|$ **then**

      $R_{best} \leftarrow R'$.

    end if

    $R \leftarrow R \setminus \{v_i\}$.

    $S \leftarrow S \setminus \{v_i\}$.

    $F \leftarrow F \cup \{v_i\}$.

    Check colours are consecutive numbers, update if not.

  end if

  $i \leftarrow i - 1$.

end while

return R_{best}.

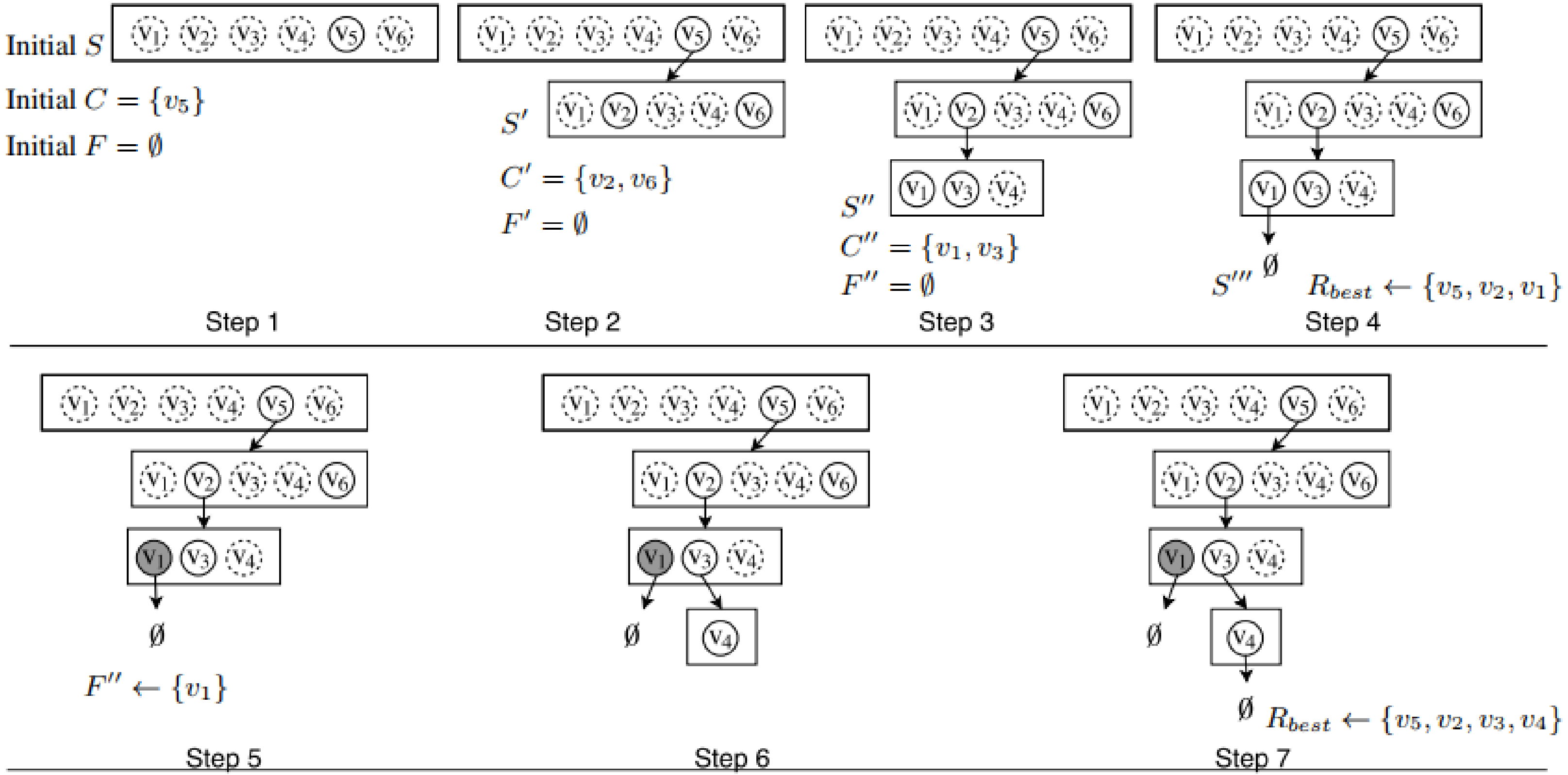


Figure 4. Progression of the search tree of Algorithm 3 when solving MC for the example graph in Fig. 2. Vertices are visited in same order that in Fig. 3, however only vertices in $C \subseteq S$ are expanded (continue circles). The first 7 steps are incrementally generated to show vertex expansions and vertex deletions (grey vertices). Leaves of the tree correspond to the empty set.

Results

- To investigate the efficacy of the proposed method, The article compared the following algorithms for matching with pairwise constraints :
- **MCQ**: BnB method (Algorithm 2)
- **PMC**: Our BnB method (Algorithm 3) with the same initial vertex ordering as MCQ.
- **MIP-MC**: MIP formulation of MC solved with GurobiOptimiser.
- **MIP-MVC**: MIP formulation of MVC solved with Gurobi Optimiser

- Two different datasets
- the Stanford 3D Scanning Repository (*armadillo*, *buddha*, *bunny*, and *dragon*)
- Mian's dataset (*chef*, *chicken*, *parasauro* and *t-rex*).

Results

Object	N	Outlier ratio	Consistency graph $\lvert V\rvert$	Consistency graph $\lvert E\rvert$	Inconsistency graph $\lvert V\rvert$	Inconsistency graph $\lvert E\rvert$	Solution $\lvert\mathcal{I}^*\rvert$	Solution angErr (°)	Solution trErr	PMC time (s)	MCQ time (s)	MIP-MC time (s)	MIP-VC time (s)	RANSAC time (s)
armadillo	1000	0.98	1000	167200	1000	831800	37	0.72	0.06	0.020	0.010	1387.404	1291.205	25.25
$\lvert\mathcal{X}\rvert = 788$	3000	0.98	3000	1397630	3000	7599370	92	0.61	0.05	2.510	5.390	–	–	39.93
$\lvert\mathcal{Y}\rvert = 711$	5000	0.98	5000	3485008	5000	21509992	139	0.98	0.15	61.080	3377.430	–	–	70.16
buddha	1000	0.96	1000	139516	1000	859484	55	0.64	0.34	0.170	0.020	75.170	85.624	5.67
$\lvert\mathcal{X}\rvert = 206$	3000	0.98	3000	700794	3000	8296206	68	1.60	0.18	5.420	2.730	–	–	105.24
$\lvert\mathcal{Y}\rvert = 193$	5000	0.99	5000	1700434	5000	23294566	74	1.06	0.25	5.640	76.100	–	–	602.04
bunny	1000	0.96	1000	101956	1000	897044	27	2.03	0.42	0.020	0.010	–	–	5.09
$\lvert\mathcal{X}\rvert = 668$	3000	0.96	3000	918160	3000	8078840	102	0.32	0.11	0.410	0.280	1224.189	1642.969	7.32
$\lvert\mathcal{Y}\rvert = 615$	5000	0.96	5000	2490092	5000	22504908	171	0.47	0.09	7.250	–	3520.973	–	16.39
chef	1000	0.94	1000	117182	1000	881818	80	1.01	0.21	0.310	0.040	16.469	22.378	1.32
$\lvert\mathcal{X}\rvert = 183$	3000	0.97	3000	774036	3000	8222964	97	0.95	0.16	1.980	1.410	1578.146	2293.338	30.39
$\lvert\mathcal{Y}\rvert = 185$	5000	0.98	5000	1961382	5000	23033618	100	0.61	0.26	6.870	37.470	–	–	200.01
chicken	1000	0.97	1000	142856	1000	856144	26	15.88	2.02	0.040	0.020	–	–	14.53
$\lvert\mathcal{X}\rvert = 601$	3000	0.98	3000	1265814	3000	7731186	55	1.63	0.28	1.480	0.830	–	–	58.08
$\lvert\mathcal{Y}\rvert = 616$	5000	0.98	5000	3597810	5000	21397190	81	0.93	0.23	7.620	21.420	–	–	98.33
dragon	1000	0.91	1000	141516	1000	857484	106	0.23	0.20	0.090	0.020	19.240	21.579	0.52
$\lvert\mathcal{X}\rvert = 289$	3000	0.97	3000	877756	3000	8119244	126	0.41	0.20	1.030	0.930	772.552	1294.577	12.74
$\lvert\mathcal{Y}\rvert = 270$	5000	0.98	5000	2211540	5000	22783460	136	0.31	0.19	5.530	18.130	–	–	76.86
parasauro	1000	0.93	1000	153806	1000	845194	81	0.14	0.10	0.040	0.020	19.053	24.703	1.26
$\lvert\mathcal{X}\rvert = 261$	3000	0.97	3000	973874	3000	8023126	118	0.40	0.10	2.830	42.160	2289.741	2681.053	21.83
$\lvert\mathcal{Y}\rvert = 216$	5000	0.98	5000	2214264	5000	22780736	127	0.44	0.22	36.600	–	–	–	84.86
t-rex	1000	0.93	1000	116406	1000	882594	86	0.43	0.15	0.040	0.010	13.601	15.705	0.88
$\lvert\mathcal{X}\rvert = 222$	3000	0.97	3000	818628	3000	8178372	118	0.13	0.21	1.200	10.570	865.289	1339.081	18.15
$\lvert\mathcal{Y}\rvert = 217$	5000	0.98	5000	2022928	5000	22972072	128	0.27	0.22	7.970	1287.740	–	–	81.86

Table 1. Results for matching with pairwise constraints (1) and RANSAC. '–' implies forced timeout after a 1-hour limit.

Analyze:

- MIP-MC took several orders of magnitude more time than PMC to find the optimal solution.
- In general, PMC found the optimal solution in less than10 seconds.
- Although MCQ performed better than PMC for *N* = 1000, PMC converged considerably faster than MCQ for *N* = 5000

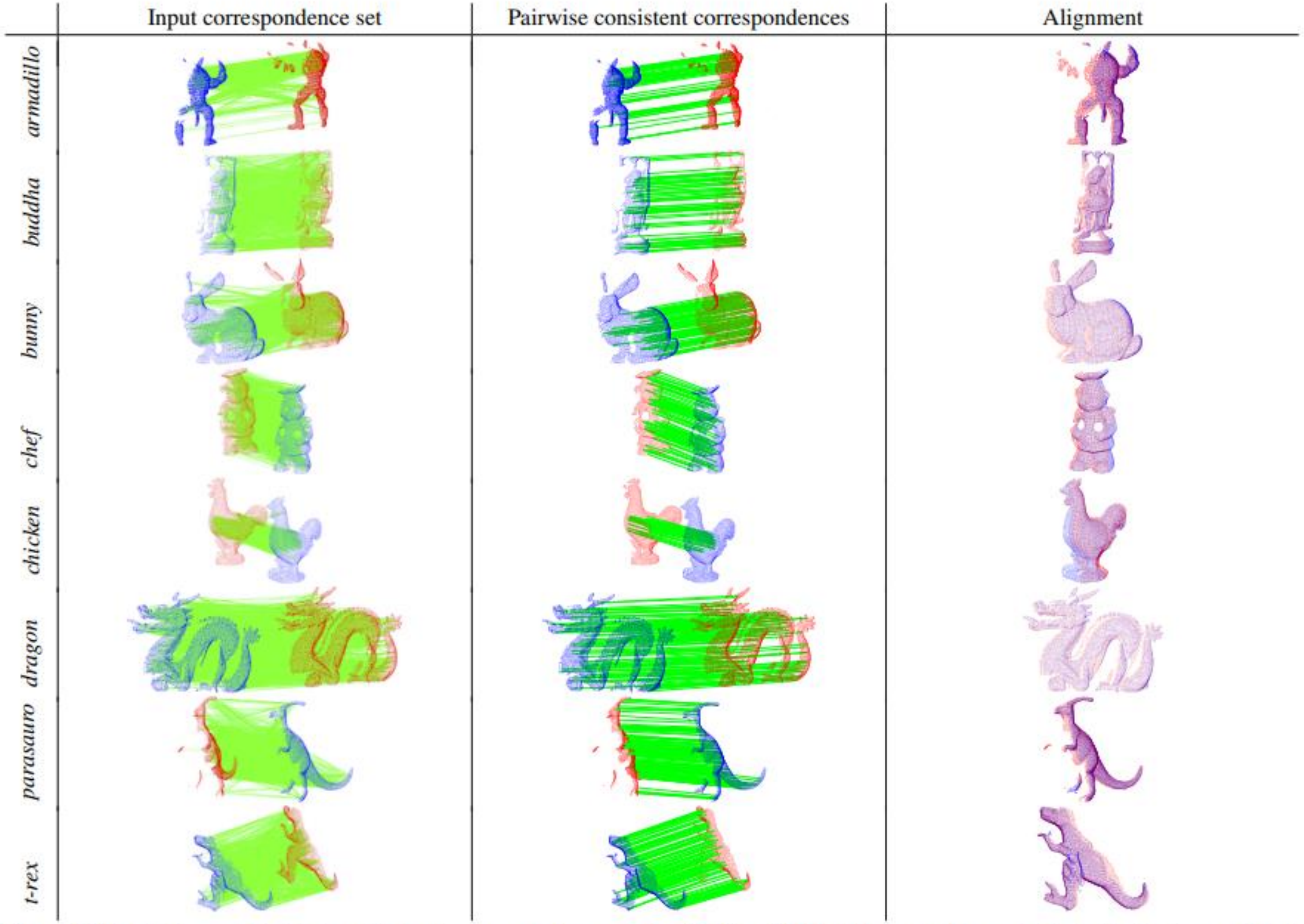


Figure 5. Qualitative results. Column 1: Input correspondence set $\mathcal{C}$ with $N = 3000$. Column 2: Largest subset of pairwise consistent correspondences $\mathcal{I}^*$. Column 3: The alignment given by the rigid transformation estimated from $\mathcal{I}^*$ using SVD [7].

conclusion

- That matching with pairwise constraints can be performed in reasonable time when posing the problem as maximum clique.
- The article have also proposed a maximum clique algorithm that combines graph colouring with a proposed extra pruning step to very efficiently solve maximum clique.
- The obtained results demonstrate that, using the proposed algorithm, matching with pairwise constraints is a very practical approach for point cloud registration

Thank You!

Avenida da Universidade, Taipa, Macau, China
Tel : (853) 8822 8833 Fax : (853) 8822 8822
Email : info@um.edu.mo Website : www.um.edu.mo