Информационные процессы, Том 24, № 1, 2024, стр. 105–117.

ТЕОРИЯ И МЕТОДЫ ОБРАБОТКИ ИНФОРМАЦИИ

Регистрация облаков точек в трехмерном пространстве с использованием мягкого соответствия [1]

А.Ю. Маковецкий[*], В.И. Кобер[,***], С.М. Воронин[*], А.В. Воронин[*], В.Н. Карнаухов[**], М.Г. Мозеров[**]**

[*] *Челябинский государственный университет, Челябинск, Россия*
e-mail: artemmac@csu.ru
[**] *Институт проблем передачи информации, Российская академия наук, Москва, Россия*
e-mail: vitaly@iitp.ru
[***] *Центр научных исследований и высшего образования, Энсенада, 22860, Мексика*

Поступила в редколлегию 28.03.2024 г. Принята 13.05.2024 г.

Аннотация—В последнее время произошел значительный прогресс в области глубокого обучения, который привел к убедительным достижениям в большинстве задач семантического компьютерного зрения, таких как классификация, обнаружение и сегментация. Регистрация облака точек — это задача, которая выравнивает два или более разных облаков точек путем оценки относительного геометрического преобразования между ними. Это хорошо известная проблема, которая играет важную роль во многих приложениях, таких как SLAM, 3D-реконструкция, картографирование, позиционирование и локализация. Сложность проблемы регистрации облака точек возрастает из-за сложности выделения признаков из-за большой разницы во внешнем виде одного и того же объекта, видимого лазерным сканером с разных точек зрения. Миллионы точек, создаваемые каждую секунду, требуют высокоэффективных алгоритмов и мощных вычислительных устройств. Известный алгоритм регистрации облака точек ICP и его варианты имеют относительно хорошую вычислительную эффективность, но, как известно, невосприимчивы к локальным минимумам и, следовательно, полагаются на качество первоначального грубого выравнивания. Работа алгоритма с помехами, вызванными зашумленными точками динамических объектов, обычно имеет решающее значение для получения удовлетворительной оценки, особенно при использовании реальных данных LiDAR. В этой статье мы предлагаем алгоритм нейронной сети для решения проблемы регистрации облака точек путем оценки мягкого соответствия между точками исходного и целевого облаков точек. Предложенный алгоритм эффективно работает с неконгруэнтными зашумленными облаками точек, генерируемыми LiDAR. Представлены результаты компьютерного моделирования, иллюстрирующие эффективность предложенного алгоритма.

КЛЮЧЕВЫЕ СЛОВА: нейронная сеть, облако точек, регистрация, реконструкция поверхностей, мягкое сопоставление.

DOI: 10.53921/18195822_2024_24_1_105

ВВЕДЕНИЕ

Регистрация трехмерных облаков точек играет решающую роль в робототехнике и компьютерном зрении, позволяя идентифицировать жесткое преобразование для выравнивания двух облаков точек с неизвестными соответствиями точек. Этот процесс широко применяется в различных областях, включая реконструкцию 3D-сцен [1]–[3], локализацию [4] и автономное вождение [5]. Традиционный итеративный метод ближайшей точки (ICP) уже давно является стандартом регистрации [6]–[14], включающий чередующиеся этапы решения соответствий

[1] Работа выполнена частично при поддержке Российского научного фонда (грант 22-19-20071)

точек и применения жестких преобразований. Однако ICP полагается на хорошее начальное выравнивание и часто сходится к локальным минимумам. В последние годы появление моделей глубокого обучения [15]–[23] произвело революцию в компьютерном зрении, что привело к созданию более быстрых и надежных алгоритмов регистрации облаков точек по сравнению с классическими методами. Несколько алгоритмов, основанных на глубоком обучении, таких как Deep Closest Point (DCP) [18], PRNet [19], RPMNet [20] и IDAM [21], используют нейронные сети для установления соответствий посредством многомерных дескрипторов. Нейронная сеть DCP, использующая Dynamic Graph CNN (DGCNN), извлекает локальные объекты из облаков точек для мягкой подгонки и решений методом наименьших квадратов, предполагая соотношение соответствия один к одному в облаках двух точек. PRNet расширяет DCP за счет включения модуля обнаружения ключевых точек для частичной регистрации. RPMNet использует отжиг для получения мягких назначений для точечных соответствий из гибридных объектов, полученных как из пространственных координат, так и из локальной геометрии. IDAM объединяет характеристики и евклидову информацию в соответствующую матрицу, используя двухэтапный метод обучения для исключения точек для регистрации.

Напротив, предлагаемый нами метод использует двухветвевую стратегию описания объектов, включающую как информацию о местоположении, так и локальные объекты, независимые от вращения. Эта стратегия направлена на достижение многомерного внедрения облаков точек, отличаясь от существующих методов, которые в значительной степени полагаются на сходство дескрипторов объектов и сталкиваются с проблемами, связанными с большими поворотами и значительными различиями в координатах облаков. В этой статье мы предлагаем простую архитектуру нейронной сети, которая использует мягкое сопоставление между точками источника и целевого облака точек и взвешенный двухточечный функционал ICP вместо стандартного двухточечного функционала ICP [6, 24]. Вес пары точек выбирается исходя из вероятности того, что пара принадлежит общим подмножествам точек облаков. Веса точек вычисляются на основе дескрипторов точек, рассчитанных в предлагаемой архитектуре. Предлагаемый алгоритм нейронной сети использует элементы сети PointNet++ [25]. Сеть обучалась на базе данных ModelNrt40 [26]. Компьютерное моделирование иллюстрирует работу предложенного алгоритма.

Статья организована следующим образом. В разделе 1 приведена формулировка задачи о описаны алгоритмы ее решения. В разделе 2 представлены результаты компьютерного моделирования. Раздел 3 содержит заключение.

1. АРХИТЕКТУРА НЕЙРОННОЙ СЕТИ

В этой статье описывается простая архитектура нейронной сети, использующая элементы сети PointNet++ [25]. Предлагаемый нейросетевой алгоритм вычисляет дескрипторы всех точек в облаках точек, определяет соответствие между точками и вычисляет преобразования из группы $SE(3)$для выравнивания двух заданных облаков точек.

Обозначим через $P = p_1, \ldots, p_{s_p}$ и $Q = q_1, \ldots, q_{s_q}$ исходное и целевое облака точек, $p_i, q_j \in R^3, i = 1, \ldots, s_p, j = 1, \ldots, s_q$. Рассмотрим точку $p_i \in P$ и ее окрестность в P радиусом r. В этой окрестности будем рассматривать K точек. Мы игнорируем дополнительные точки, если окрестность содержит более K точек, и добавляем копии точки из окрестности, если окрестность содержит менее K точек. Таким образом, мы сопоставляем набор, содержащий K элементов, точке p_i и делаем это для всех точек P и Q. Дескрипторы всех точек в P вычисляются следующим образом.

Обозначим через B размер батча.

Первый слой. Исходный тензор имеет размер $(B, 3, s_p)$. Используем ранее вычисленные окрестности всех точек s_p и получаем тензор $(B, s_p, K, 3)$, после перестановки размерностей

тензор имеет размер $(B, 3, K, s_p)$. Мы используем 2D-свертку с параметрами $in\_channels = 3, out\_channels = 64, kernel\_size = 1, stride = 1, padding = 0$, т.е. тензор после свертки имеет размер $(B, 64, K, s_p)$. Используется функция активации ReLu и нормализация батча. После свертки вычисляется максимум в тензоре вдоль размерности с индексом 2 (размерность имеет размер K) и на выходе получается тензор с параметрами $(B, 64, s_p)$.

Третий слой. Входной тензор имеет размер $(B, 128, s_p)$. Используем ранее вычисленные окрестности всех точек s_p и получаем тензор $(B, s_p, K, 128)$, после перестановки размерностей тензор имеет размер $(B, 128, K, s_p)$. Мы используем 2D-свертку с параметрами $in\_channels = 128, out\_channels = 256, kernel\_size = 1, stride = 1, padding = 0$, т.е. тензор после свертки имеет размер $(B, 256, K, s_p)$. Используется функция активации ReLu и нормализация батча. После свертки находим максимум в тензоре вдоль размерности с индексом 2 (размерность имеет размер K) и получаем тензор с параметрами $(B, 256, s_p)$.

Четвертый слой. Исходный тензор имеет размер $(B, 256, s_p)$. Используем ранее вычисленные окрестности всех точек s_p и получаем тензор $(B, s_p, K, 256)$, после перестановки размерностей тензор имеет размер $(B, 256, K, s_p)$. Мы используем 2D-свертку с параметрами $in\_channels = 256, out\_channels = 256, kernel\_size = 1, stride = 1, padding = 0$, т.е. тензор после свертки имеет размер $(B, 256, K, s_p)$. Используется функция активации ReLu и нормализация батча. После свертки находим максимум в тензоре вдоль размерности с индексом 2 (размерность имеет размер K) и получаем тензор с параметрами $(B, 256, s_p)$.

Пятый слой. Исходный тензор имеет размер $(B, 256, s_p)$. Используем ранее вычисленные окрестности всех точек s_p и получаем тензор $(B, s_p, K, 256)$, после перестановки размерностей тензор имеет размер $(B, 256, K, s_p)$. Мы используем 2D-свертку с параметрами $in\_channels = 256, out\_channels = 512, kernel\_size = 1, stride = 1, padding = 0$, т.е. тензор после свертки имеет размер $(B, 512, K, s_p)$. Используется функция активации ReLu и нормализация батча. После свертки находим максимум в тензоре вдоль размерности с индексом 2 (размерность имеет размер K) и получаем тензор с параметрами $(B, 512, s_p)$.

Вектор размера 512 рассматривается как дескриптор точки P. Аналогичным образом мы вычисляем дескрипторы точек в облаке точек Q. Когда дескрипторы всех точек в P и Q вычислены, мы генерируем матрицу W размера $s_p \times s_q$. Элемент матрицы W_{ij} являются вероятностями наличия соответствия между точками p_i и q_j, $i = 1, \ldots, s_p$, $j = 1, \ldots, s_q$. Матрица W используется для формирования виртуального облака точек $(vQ) = (vq)_1, \ldots, (vq)_{s_p}$ следующим образом:

$$(vq)_i = \sum_{j=1}^{s_q} W_{ij} q_j, \tag{1}$$

где $i = 1, \ldots, s_p$. Предположим, что точка p_i соответствует точке $(vq)_i, i = 1, \ldots, s_p$. Таким образом, мы задаем мягкое сопоставление между облаками точек P и Q. Далее происходит поиск максимального значения w_i в i-й строке матрицы W. Значение считается весовым коэффициентом w_i пары $(p_i, (vq)_i), i = 1, \ldots, s_p$. Рассмотрим следующий функционал:

$$J(R, T) = \sum_{i=1}^{s_p} w_i ||Rp_i + T - (vq)_i||^2, \tag{2}$$

где $R \in SO(3)$, $T \in R^3$ и условную вариационную задачу

$$(R_*, T_*) = arg \min_{(R,T)} J(R, T), \tag{3}$$

при условии, что $R \in SO(3)$. Обозначим через C_p центр масс облака P

$$C_p = (1/\sum_{i=1}^{s_p} w_i) \sum_{j=1}^{s_p} p_j, \tag{4}$$

и центроид P' облака P

$$p'_i = p_i - C_p, \tag{5}$$

где $i = 1, \ldots, s_p$. Обозначим через $\mathbf{P'}$ матрицу центроида P

$$\mathbf{P'} = \begin{pmatrix} p'_{11} & p'_{s_p 1} \\ p'_{12} \ldots & p'_{s_p 2} \\ p'_{13} & p'_{s_p 3} \end{pmatrix}. \tag{6}$$

Аналогично обозначим матрицу $\mathbf{(vQ)'}$ центроида $(vQ)'$

$$\mathbf{P'} = \begin{pmatrix} (vq)'_{11} & (vq)'_{s1} \\ (vq)'_{12} \ldots & (vq)'_{s2} \\ (vq)'_{13} & (vq)'_{s3} \end{pmatrix}. \tag{7}$$

Также обозначим через $\mathbf{P''}$ следующую матрицу:

$$\mathbf{P''} = \begin{pmatrix} w_1 p'_{11} & w_{s_p} p'_{s_p 1} \\ w_1 p'_{12} \ldots & w_{s_p} p'_{s_p 2} \\ w_1 p'_{13} & w_{s_p} p'_{s_p 1} \end{pmatrix}. \tag{8}$$

Вариационная задача (3) может быть сведена к условной вариационной задаче

$$R_* = arg \max_R < R\mathbf{P''}, (\mathbf{vQ})' >, \tag{9}$$

при условии, что $R \in SO(3)$. Опишем матрицу M

$$M = (\mathbf{vQ})'(\mathbf{P''})^t. \tag{10}$$

Матрица M может быть представлена как матричное произведение с помощью разложения по сингулярным значениям

$$M = USV^t, \tag{11}$$

где U и V^t — ортогональные матрицы, а S — диагональная матрица. Отметим, что PyTorch поддерживает SVD разложение и обратное распространение ошибки для него. Решение вариационной задачи (9) принимает вид

$$R_* = \begin{cases} UV^t, & \text{if } det(U)det(V^t) = 1 \\ Udiag(1,1,-1)V^t\ , & \text{if } det(U)det(V^t) = -1 \end{cases}, \tag{12}$$

и вектор параллельного переноса можно вычислить как

$$T_* = (1/\sum_{i=1}^{s_p} w_i) \sum_{j=1}^{s_p} w_j((vq)_j - R_* p_j). \tag{13}$$

Число обучаемых весов полученной сети составляет 480344. Функция ошибок представляет собой взвешенную сумму MSE между оцененной и истинной матрицами преобразования (вращения и перемещения).

Использовался алгоритм Adam с параметром скорости обучения 0,001.

Размер батча B=10 и размер окрестности K=32. Сеть обучалась по парам облаков точек из части базы данных ModelNet40 [26].

Сеть обучалась в течение 5 эпох. Значение функции потерь loss=0,66 до обучения, loss=0,0339 после 1-й эпохи, loss=0,0114 после 2-й эпохи, loss=0,0081 после 3-й эпохи, loss=0,0076 после 4-й эпохи , потеря=0,0075 после 5-й эпохи.

2. КОМПЬЮТЕРНОЕ МОДЕЛИРОВАНИЕ

Мы применяем предложенную нейронную сеть к базе данных ModelNet40 [26]. База данных содержит 12311 сетчатых CAD-моделей, относящихся к 40 классам. Обучающая часть базы данных состоит из 9843 облаков точек, а тестовая часть — из 2468 облаков точек. Для каждой модели с граней сетки равномерно выбираются 1024 точки. Используется только информация из базы данных о координатах точки. Для каждого облака $P = \{p_1, \ldots, p_{1024}\}$ случайно выбираются углы поворота относительно координатных осей x, y и z в диапазоне $[-\pi/8; \pi/8]$ и случайно выбранные координаты вектора перемещения с его координатами в диапазоне $[-0,25; 0,25]$. Матрица (в однородных координатах) данного преобразования обозначается как M_{true}. Мы применяем матрицу M_{true} к матрице $\mathbf{P}$, связанной с облаком P, и получаем матрицу $\mathbf{Q}$, связанную с облаком $Q = \{q_1, \ldots, q_{1024}\}$

$$\mathbf{Q} = M_{true}\mathbf{P}. \tag{14}$$

Мы используем предложенную нейронную сеть (CNN) в качестве алгоритма грубого выравнивания облаков точек и point-to-point ICP (PtP) в качестве алгоритма уточнения. Для демонстрации эффективности предложенного метода были выполнены 10 экспериментов, результаты которых показаны на следующих рисунках и таблицах.

На Рис.1(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.1(b) показано положение облаков точек после CNN, на Рис.1(c) показано положение облаков точек после CNN+PtP. Таблица 1 показывает ошибки рассмотренных ал-

(a) (b) (c)

Рис. 1. (a) — Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

горитмов относительно норм L_1 и L_2.

Таблица 1. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	0.949362	0.165098	0.000002	0.000000
L_2	0.376923	0.101902	0.000001	0.000000

На Рис.2(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.2(b) показано положение облаков точек после CNN, на Рис.2(c) показано положение облаков точек после CNN+PtP. Таблица 2 показывает ошибки рассмотренных ал-

(a) (b) (c)

Рис. 2. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

горитмов относительно норм L_1 и L_2.

Таблица 2. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

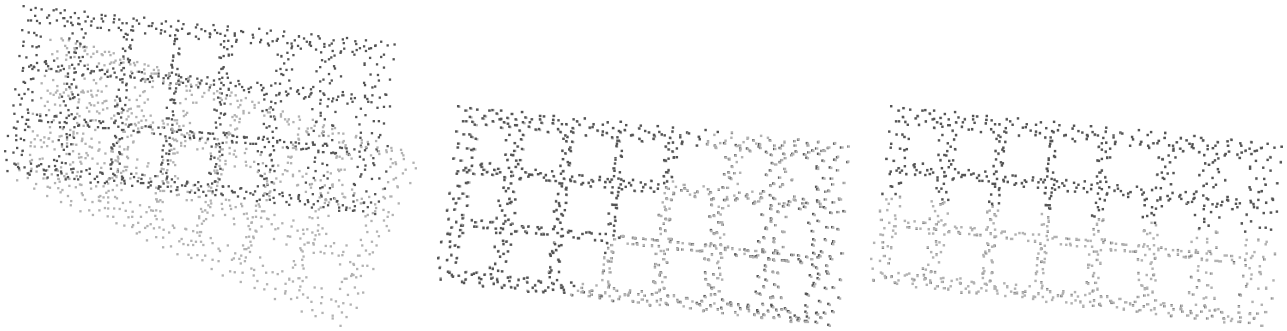
	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	0.505952	0.080382	0.000002	0.000000
L_2	0.227929	0.059801	0.000001	0.000000

На Рис.3(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.3(b) показано положение облаков точек после CNN, на Рис.3(c) показано положение облаков точек после CNN+PtP. Таблица 3 показывает ошибки рассмотренных алгоритмов относительно норм L_1 и L_2.

Таблица 3. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	0.059883	0.005147	0.000002	0.000000
L_2	0.028952	0.004066	0.000001	0.000000

На Рис.4(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.4(b) показано положение облаков точек после CNN, на Рис.4(c) показано положение облаков точек после CNN+PtP. Таблица 4 показывает ошибки рассмотренных алгоритмов относительно норм L_1 и L_2.



(a) (b) (c)

Рис. 3. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

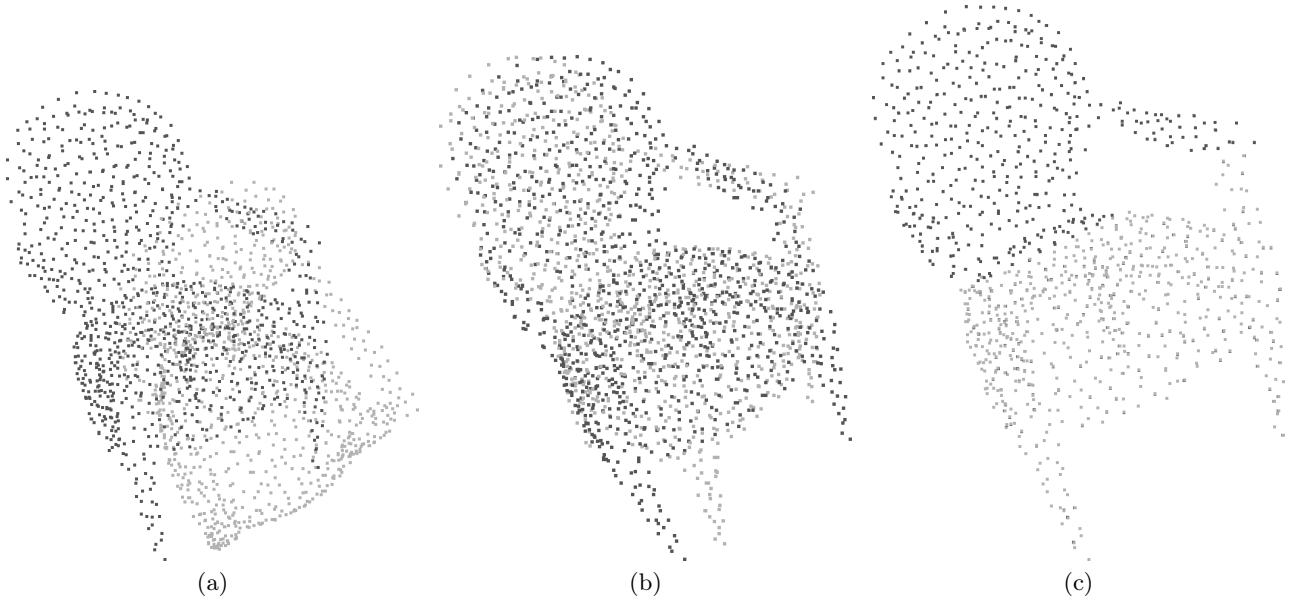
(a) (b) (c)

Рис. 4. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

Таблица 4. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	0.376001	0.121047	0.000001	0.000000
L_2	0.167898	0.071043	0.000001	0.000000

На Рис.5(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.5(b) показано положение облаков точек после CNN, на Рис.5(c) показано положение облаков точек после CNN+PtP. Таблица 5 показывает ошибки рассмотренных алгоритмов относительно норм L_1 и L_2.



(a) (b) (c)

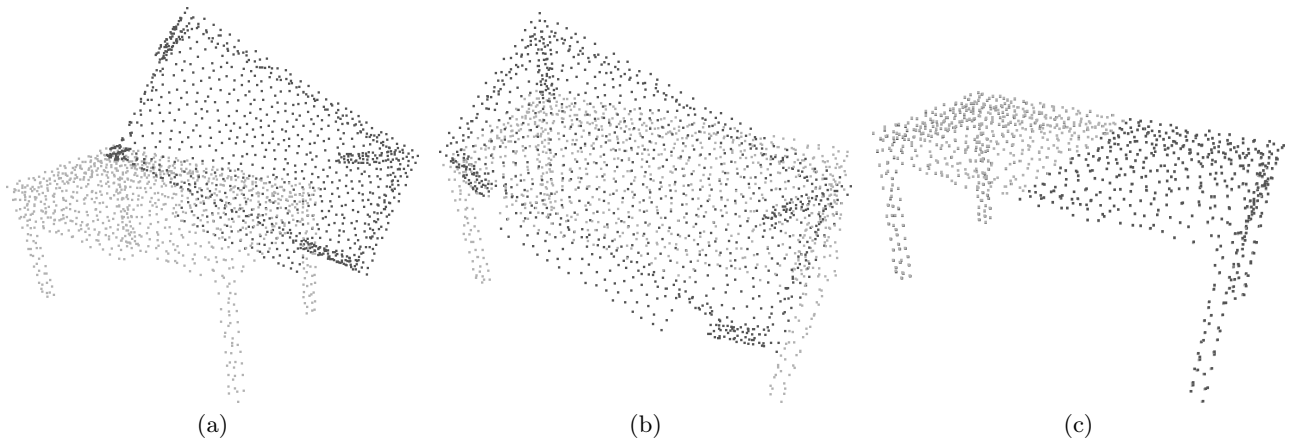
Рис. 5. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

gh!]

Таблица 5. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	1.948246	0.304671	0.000001	0.000000
L_2	1.948246	0.187354	0.000001	0.000000

На Рис.6(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.6(b) показано положение облаков точек после CNN, на Рис.6(c) показано положение облаков точек после CNN+PtP. Таблица 6 показывает ошибки рассмотренных ал-



(a) (b) (c)

Рис. 6. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

горитмов относительно норм L_1 и L_2.

Таблица 6. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	2.275989	0.249078	0.000003	0.000001
L_2	0.933218	0.160645	0.000001	0.000000

На Рис.7(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.7(b) показано положение облаков точек после CNN, на Рис.7(c) показано положение облаков точек после CNN+PtP. Таблица 7 показывает ошибки рассмотренных ал-

(a) (b) (c)

Рис. 7. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

горитмов относительно норм L_1 и L_2.

Таблица 7. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

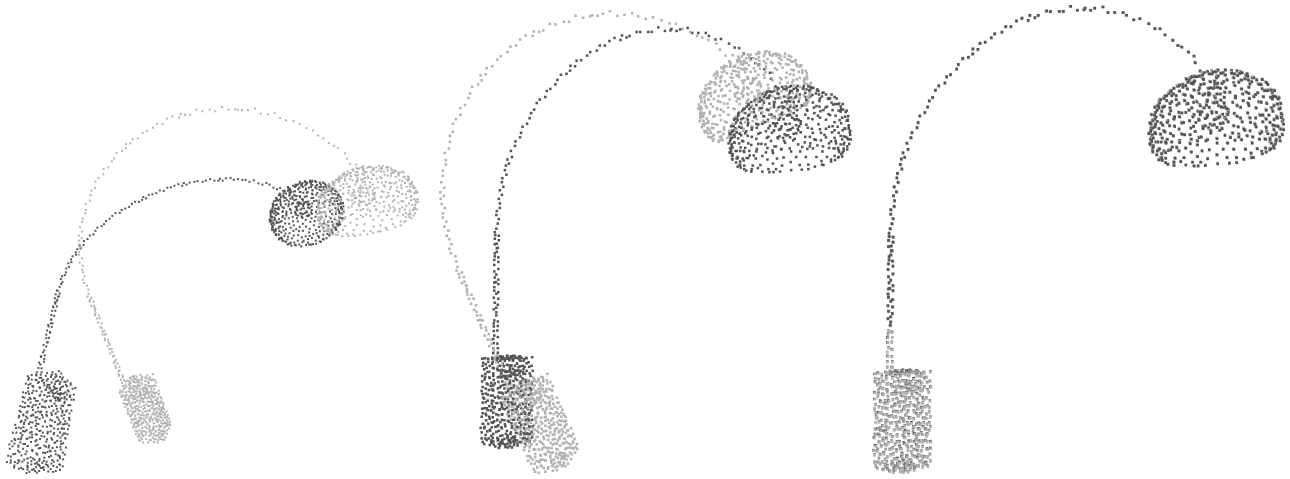
	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	0.942749	0.160958	0.000002	0.000001
L_2	0.367263	0.109005	0.000001	0.000000

На Рис.8(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.8(b) показано положение облаков точек после CNN, на Рис.8(c) показано положение облаков точек после CNN+PtP. Таблица 8 показывает ошибки рассмотренных алгоритмов относительно норм L_1 и L_2.

Таблица 8. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	1.808907	0.174372	0.000001	0.000000
L_2	0.689345	0.111230	0.000000	0.000000

На Рис.9(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.9(b) показано положение облаков точек после CNN, на Рис.9(c) показано положение облаков точек после CNN+PtP. Таблица 9 показывает ошибки рассмотренных ал-



(a) (b) (c)

Рис. 8. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

(a) (b) (c)

Рис. 9. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) —результат регистрации CNN; (c) — результат регистрации CNN+PtP.

Таблица 9. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	1.198202	0.270670	0.000001	0.000000
L_2	0.541937	0.167172	0.000001	0.000000

горитмов относительно норм L_1 и L_2.

На Рис.10(a) показано начальное положение исходного облака точек P и целевого облака точек Q, на Рис.10(b) показано положение облаков точек после CNN, на Рис.10(c) показано положение облаков точек после CNN+PtP. Таблица 10 показывает ошибки рассмотренных алгоритмов относительно норм L_1 и L_2.

(a) (b) (c)

Рис. 10. (a) —Исходное облако точек (серые точки) и целевое облако точек (черные точки); (b) — результат регистрации CNN; (c) — результат регистрации CNN+PtP.

Таблица 10. Точность вычисления поворота и параллельного переноса относительно норм L_1 и L_2.

	R_{CNN}	T_{CNN}	$R_{CNN+ICP}$	$T_{CNN+ICP}$
L_1	7.443286	1.626208	7.383439	1.719010
L_2	2.778001	1.059065	2.812945	1.072411

Мы применяем большие углы поворота в эксперименте № 10, и алгоритм возвращает ложное преобразование, поскольку предлагаемая CNN обучена на достаточно малых углах поворота.

3. ЗАКЛЮЧЕНИЕ

В этой статье мы предложили простой алгоритм нейронной сети для регистрации облаков точек в трехмерном пространстве. Предлагаемая нейронная сеть основана на мягком сопоставлении между исходным и целевым облаками точек и использует взвешенную функцию ICP от точки к точке для оценки ортогонального преобразования, которое выравнивает облака точек. Мягкое сопоставление облаков точек с использованием матрицы вероятностей описывает семантическое сходство точек. Предложенная сеть совместно с двухточечным ICP показывает хорошие результаты в задаче реконструкции 3D-модели. Эффективность предложенного алгоритма оценивается в базе данных ModelNet40.

СПИСОК ЛИТЕРАТУРЫ

1. Izadi, S., Kim, D., Hilliges, O., Molyneaux, D., Newcombe, R., Kohli, P., Shotton, J., Hodges, S., Freeman, D., Davison, A., et al. Kinectfusion: Real-time 3d reconstruction and interaction using a moving depth camera. // Proc. of the ACM Symposium on Utilizer Interface Software and Technology, 559–568 (2011).

2. Eldefrawy, M., King, S. and Starek, M. Partial Scene Reconstruction for Close Range Photogrammetry Using Deep Learning Pipeline for Region Masking. // Remote Sens. 14, 3199 (2022).

3. Zhang, Z., Dai, Y. and Sun, J. Deep learning based point cloud registration: An overview. // Virtual Real. Intell. Hardw., 2, 222–246, (2020).

4. Chen, K., Lopez, B., Agha-Mohammadi, A. and Mehta, A. Direct lidar odometry: Fast localization with dense point clouds. // IEEE Robot. Autom. Lett., 7, 2000–2007 (2022).

5. Zheng, Y., Li, Y., Yang, S. and Lu, H. Global-PBNet: A novel point cloud registration for autonomous driving. // IEEE Trans. Intell. Transp. Syst., 23, 22312–22319 (2022).

6. Besl, P. and McKay, N. A method for registration of 3-D shapes. // IEEE Transactions of Pattern Analysis and Machine Intelligence, 2, 239–256 (1992).

7. Chen, Y. and Medioni, G. Object modeling by registration of multiple range images. // Image and Vision Computing, 10, 145–155, (1992).

8. Horn, B., Hilden, H. and Negahdaripour, S. Closed-form Solution of Absolute Orientation Using Orthonormal Matrices.// Journal of the Optical Society of America, Series A, 5, 7, 1127–1135 (1988).

9. Umeyama, S. Least-squares estimation of transformation parameters between two point patterns. // IEEE-TPAMI, 13(4), 376–380 (1991).

10. Makovetskii, A., Voronin, S., Kober, V. and Voronin, A. An efficient algorithm for non-rigid object registration. // Computer Optics, 44, 67–73 (2020).

11. Makovetskii, A., Voronin, S., Kober, V. and Voronin, A. A non- iterative method for approximation of the exact solution to the point-to- plane variational problem for orthogonal transformations. // Mathematical Methods in the Applied Sciences, 41(18), 9218–9230 (2018).

12. Makovetskii, A., Voronin, S., Kober, V. and Voronin, A. A regularized point cloud registration approach for orthogonal transformations. // Journal of Global Optimization (2020).

13. Makovetskii, A., Voronin, S., Kober, V. and Voronin, A. Point Cloud Registration Based on Multiparameter Functional. // Mathematics, 9, 2589 (2021).

14. Makovetskii, A., Voronin, S., Kober, V. and Voronin, A. Coarse Point Cloud Registration Based on Variational Functional. // Mathematics, 11, 35 (2023).

15. Aoki, Y., Goforth, H., Srivatsan, R. and Lucey, S. Pointnetlk: Robust and efficient point cloud registration using pointnet. // Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 7163–7172 (2019).

16. Sarode, V., Li, X., Goforth, H., Aoki, Y., Srivatsan, R., Lucey, S. and Choset, H. Pcrnet: Point cloud registration network using pointnet encoding. // arXiv:1908.07906 (2019).

17. Xu, H., Liu, S., Wang, G., Liu, G. and Zeng, B. Omnet: Learning overlapping mask for partial-to-partial point cloud registration. // Proc. of the IEEE/CVF International Conference on Computer Vision, 3132–3141 (2021).

18. Wang, Y. and Solomon, J. Deep closest point: learning representations for point cloud registration. // International Conference on Computer Vision (ICCV), IEEE, 3522–3531 (2019).

19. Wang, Y. and Solomon, J. Prnet: Self-supervised learning for partial-to-partial registration. // arXiv:1910.12240, (2019).

20. Yew, Z. and Lee, G. Rpm-net: Robust point matching using learned features. // Proc. of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 11824–11833 (2020).

21. Li, J., Zhang, C., Xu, Z., Zhou, H. and Zhang, C. Iterative distance-aware similarity matrix convolution with mutual-supervised point elimination for efficient point cloud registration. // Proc. of the 16th European Conference on Computer Vision, 378–394 (2020).

22. Lu, W., Zhou, Y., Wan, G., Hou, S. and Song. S. L3-Net: Towards learning based LiDAR localization for autonomous driving. // Proc. of the IEEE Conference on Computer Vision and Pattern Recognition, (2019).

23. Voronin, S., Vasilyev, A., Kober, V., Makovetskii, A., Voronin, A. and Zhernov, D. Neural network for 3D point clouds alignment. // Proc. SPIE 12226, Applications of Digital Image Processing XLV, 122261H, (2022).

24. Voronin, S., Vasilyev, A., Kober, V., Makovetskii, A., Voronin, A., Zhernov, D. (2023, October). Deep neural network for incongruent point clouds registration. In Applications of Digital Image Processing XLVI (Vol. 12674, pp. 409–419). SPIE.

25. Qi, C. R., Yi, L., Su, H., Guibas, L. J. (2017). Pointnet++: Deep hierarchical feature learning on point sets in a metric space. Advances in neural information processing systems, 30.

26. Z. Wu, S. Song, A. Khosla, F. Yu, L. Zhang, X. Tang, and J. Xiao. 3D Shapenets: A Deep Representation for Volumetric Shapes. // Proc. CVPR, pp. 1912–1920, 2015.

Point cloud registration in 3D space using soft matching

A. Makovetskii, V. Kober, S. Voronin, A. Voronin, V. Karnaukhov, M. Mozerov

Recently, there has been significant progress in the field of deep learning, which has led to compelling advances in most semantic computer vision tasks such as classification, detection, and segmentation. Point cloud registration is a task that aligns two or more different point clouds by estimating the relative geometric transformation between them. This is a well-known problem that plays an important role in many applications such as SLAM, 3D reconstruction, mapping, positioning, and localization. The complexity of the point cloud registration problem increases due to the difficulty of feature extraction due to the large difference in the appearance of the same object as seen by a laser scanner from different points of view. The millions of points created every second require highly efficient algorithms and powerful computing devices. The well-known ICP point cloud registration algorithm and its variants have relatively good computational efficiency, but are known to be immune to local minima and therefore rely on the quality of the initial coarse alignment. Dealing with the interference caused by noisy points on dynamic objects is usually critical to obtaining a satisfactory estimate, especially when using real LiDAR data. In this paper, we propose a neural network algorithm to solve the point cloud registration problem by estimating the soft correspondence between the points of the source and target point clouds. The proposed algorithm works effectively with incongruent noisy point clouds generated by LiDAR. The results of computer simulation are presented, illustrating the effectiveness of the proposed algorithm.

KEYWORDS: neural network, point cloud, registration, shape analysis, soft matching.