2024 统计信号处理实验大作业

中国科学院大学

电子电气与通信工程学院

姓名：李厚华

学号：202418019427056

1. 实验背景

随着智能家居和安防技术的发展，毫米波雷达因其高精度、高分辨率和隐私保护等优点，广泛应用于人体检测与行为识别。毫米波雷达能够在复杂环境中准确捕捉人体的运动信息，特别适用于监测无人存在、自由运动以及特定动作（如坐下摔倒等）的场景。本实验旨在通过处理 TI IWR6843 毫米波雷达采集的数据，完成对目标的检测、轨迹提取及速度估算，为智能监控和安全预警系统提供数据支持。

2. 理论背景

2.1. 数据预处理

1. 数据读取：

通过 MATLAB 实现，主要功能是从 JSONL 文件中逐行读取数据，并解析每一帧的的数据，将所有数据均以 cell 形式保存，确保每一帧的数据点可以独立存储和处理。存储数据如下：

- frames 是一个 cell 数组，存储每一帧的帧号。
- point_clouds 是一个 cell 数组，存储每一帧的点云数据，每个元素是一个 n×3 的矩阵。
- targets 是一个 cell 数组，存储每一帧的目标坐标，每个元素是一个 1×3 的向量。

2. 数据清洗：

- 基于 targets 的全局坐标范围，寻找真值的最大值与最小值，然后向上和向下取整划定数据的正常范围，确保过滤后的点云数据与目标数据在空间上保持一致，通过过滤不符合范围的点云数据，去除噪声和异常值，提高数据质量。
- 同时保持输出数据的维度与输入一致，使得相应帧数据的时间相对关系不变，便于后续

的数据处理和分析，比如计算速度和中心点云。

2.2. 信号点与噪声点的区分

在本实验中，信号点与噪声点的区分通过两个步骤实现：首先使用阈值筛选有效帧数据，然后基于密度的去噪算法去除点云中的噪声数据。以下是该部分的理论总结：

1. 阈值筛选有效帧数据：

通过设置最小点数阈值（threshold），本实验设置的默认数值是 10，过滤掉点数过少的帧数据，确保后续处理的点云数据具有足够的密度和信息量

实现：遍历每一帧点云数据，检查当前帧的点数是否大于等于 threshold。如果点数少于 threshold，则将该帧数据标记为空（[]），跳过后续处理。如果点数满足要求，则继续基于密度的去噪处理。

2. 基于密度的去噪算法

通过计算每个点的邻域密度，区分信号点和噪声点。信号点通常位于高密度区域，而噪声点则位于低密度区域[1,2]。

实现：以每个点为中心，半径为 radius 的球形邻域确定为该点的邻域范围，因为本实验是对人体点云数据的筛选，所以设置的半径为 0.5m。然后统计每个点邻域内的点数（不包括自身），如果某个点的邻域点数小于 min_neighbors，则判定为噪声点，如果邻域点数大于等于 min_neighbors 的点，作为有效信号点。

2.3. 目标的三维运动轨迹提取

从过滤后的点云数据中提取目标的真实位置，并按照时间关系推断物体的运动轨迹。

实现：通过计算每一帧及其前后各 n 帧（共 2n+1 帧）的非空点云数据的均值，作为当前帧的平均位置，从而生成平滑的轨迹数据。如果某帧的点云数据为空，则跳过该帧并

仅计算剩余非空数据的均值。该方法能够有效减少噪声对轨迹提取的影响，同时保留目标的运动趋势，为目标跟踪和运动分析提供了可靠的基础。

2.4. 目标运动速度估算

目标速度的估计基于去除噪声后的点云数据提取的轨迹信息。首先，通过计算每个点云数据的平均位置计算速度变化量（存储在 trajectory 中）。随后，利用帧数据之间的相对时间关系计算时间间隔，最后计算目标的速度。

实现：通过遍历轨迹点，计算相邻非空点之间的位移，并结合给定的时间间隔（默认55ms*帧间隔）计算速度。对于缺失数据的帧，函数通过前后非空点的速度进行插值（本实验直接由平均速度代替），确保速度曲线的连续性。最终，输出每个轨迹点对应的速度值speeds，单位为米/秒。

3. 结果分析与讨论

3.1. 信号与噪声点区分效果

基于 2.2 节的噪声剔除算法，我们实现了函数 **[filtered_point_clouds, noise_point_clouds] = filter_noise(point_clouds, threshold, radius, min_neighbors)**。该函数中，point_clouds 表示经过异常值剔除后的点云数据，threshold 为可用帧点云数量的阈值，radius 为邻域半径（本实验中设置为 0.5 米），通过欧氏距离与阈值进行比较，min_neighbors 为最小邻居数，用于判定某一点是否为有效信号点。

为验证算法的有效性，我们从三个点云数据中随机选取了 4 帧，其噪声滤除效果如下：

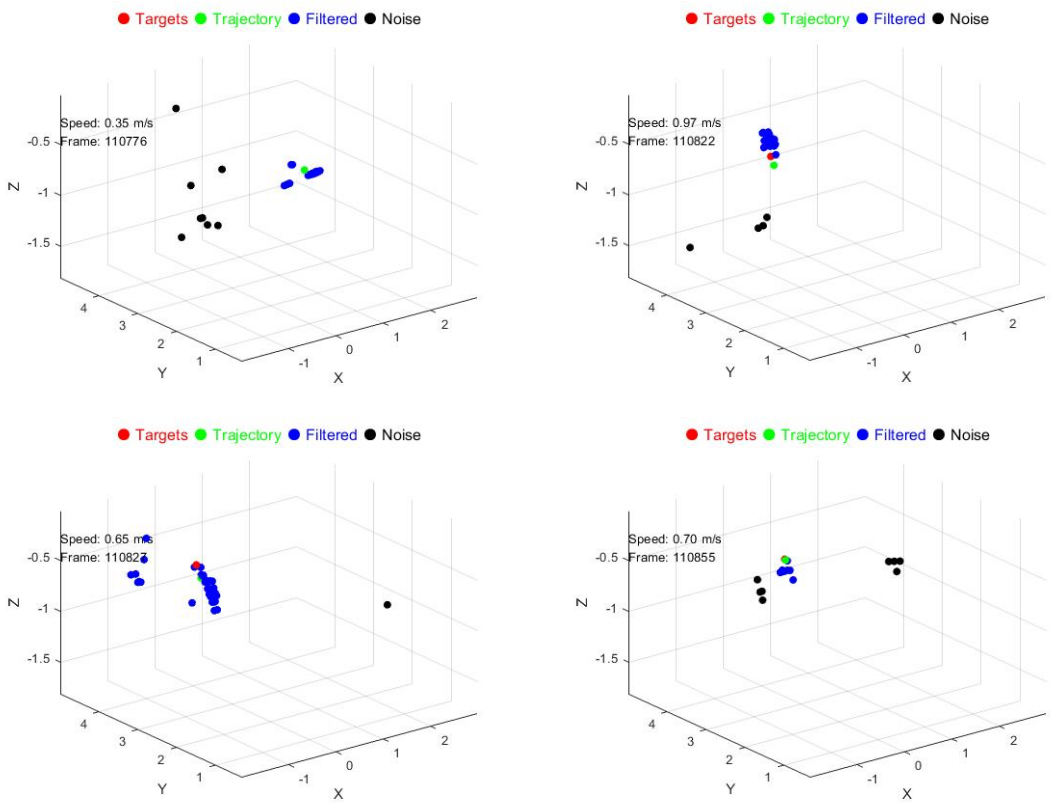

图 1 信号点与噪声点区分图

从结果图中可以清晰地观察到，经过过滤后的有效信号点分布较为集中，且与真值（Targets）的吻合度较高。偏离真值的点均被准确地判定为噪声点，这表明去噪算法在有效保留目标信息的同时，能够高效地去除噪声。这一结果验证了去噪算法在目标提取和噪声抑制方面的有效性，为后续的目标跟踪和运动分析提供了可靠的数据基础。

3.2. 运动轨迹的准确性

基于 2.3 节的运动轨迹提取算法，我们实现了函数 **trajectory = extract_trajectory(filtered_point_clouds)**。该函数中，`filtered_point_clouds` 为经过噪声剔除后的点云数据，`trajectory` 为提取的估计运动轨迹。我们对三个文件的点云数据进行了处理，提取了相应的运动轨迹，并将其与真实值（Targets）进行了对比，实验结果如下：

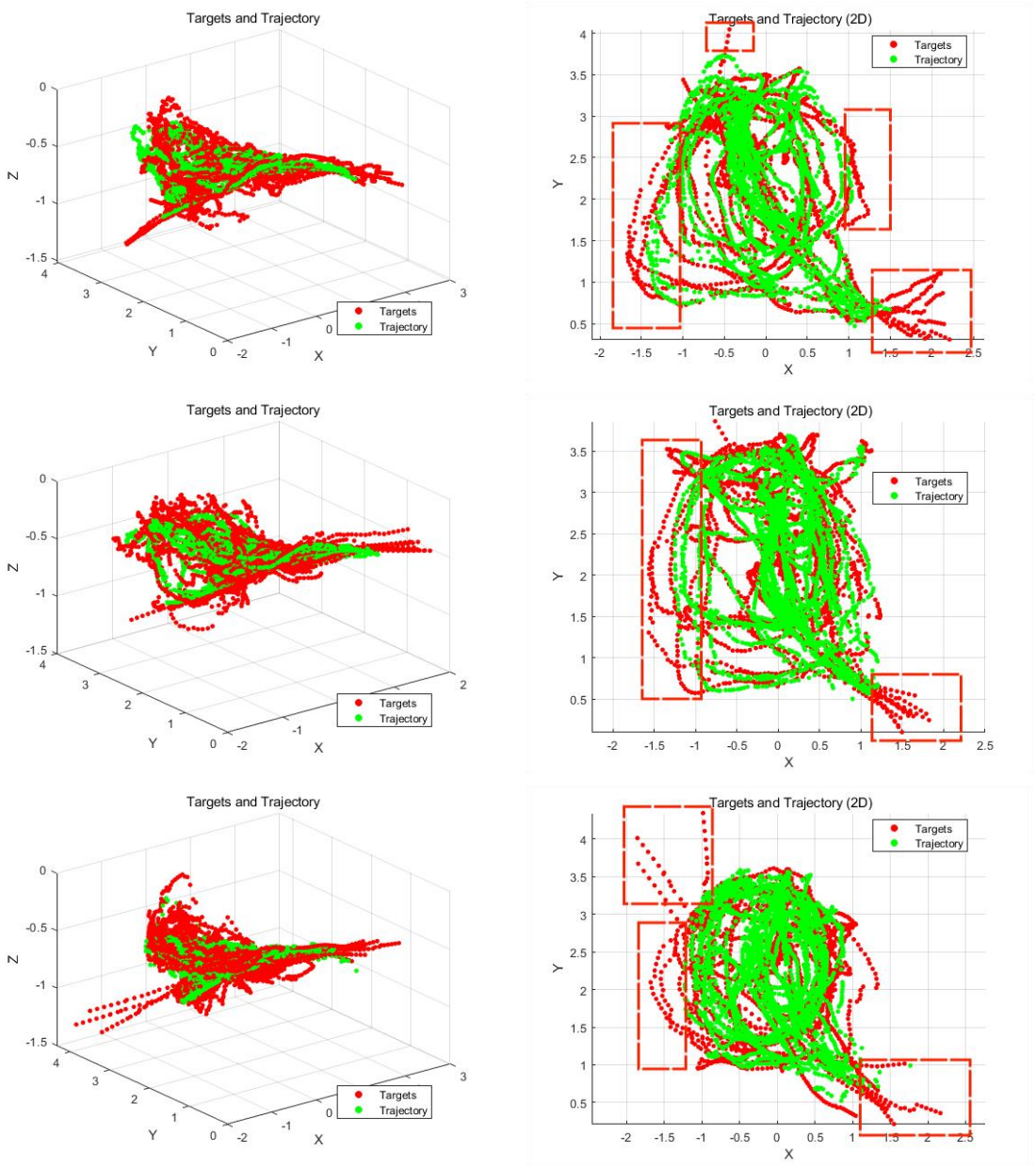

图 2 运动轨迹图

注：从上到下依次是 data_1c、data_2c 和 data_4z 数据的 Trajectory 与真值 Targets 三维与 xy 二维图像。

从实验结果中可以观察到，在中心区域，运动轨迹提取结果（Trajectory）与真值（Targets）的重合度较高，表明算法在该区域具有较好的性能。然而，在运动轨迹的边缘区域，实验结果与真值之间存在显著偏差，这一现象在图 2 右侧三个子图的标注位置尤为明显。分析其原因，可能包括以下两个方面：

1. 均值滤波的平滑效应：在边缘区域，均值滤波对数据产生了较强的平滑作用，导致

边缘细节信息丢失，从而影响了轨迹提取的精度。

2. 毫米波雷达采集范围限制：由于毫米波雷达的采集范围有限，边缘区域的点云数据可能采集不足，导致轨迹提取算法在边缘区域的性能下降。

这一分析为后续算法优化提供了重要依据，特别是在边缘区域的数据处理和改进采集技术方面。

为了量化分析运动轨迹的偏差，本实验实现了函数 **avg_distance = compute_avg_euclidean_distance(targets, trajectory)**，用于计算目标数据 Targets 与估计轨迹数据 Trajectory 之间的平均欧式距离偏差。该函数通过遍历两组数据，筛选出对应元素均非空的情况，计算每一对点的欧式距离，并最终求取平均值以量化轨迹偏差。实验结果如表 1 所示：

表 1 运动轨迹偏差

数据	data_1c	data_2c	data_4z
欧式距离偏差 /m	0.1898	0.1685	0.2300

根据表格数据，可以得出以下结论：data_4z 数据的欧式距离偏差最大，data_1c 数据的欧式距离偏差次之，而 data_2c 数据的欧式距离偏差最小。这一现象在图像中得到了直观的验证，表明不同数据集在空间偏差上存在显著差异。

3.3. 速度估算的准确性

基于 2.4 节的目标运动速度计算方法，我们实现了函数 [speeds] = estimate_speed(trajectory, time_interval)。该函数中，trajectory 为 3.2 节中提取的估计运动轨迹，time_interval 为默认时间间隔，其值为 55 毫秒。该函数通过计算相邻轨迹点之间的位

移变化并结合时间间隔，估计目标的运动速度。

对三个文件的点云数据提取的运动轨迹分别估计运动速度，并与真值 Targets 估计的速度进行对比，得到的运行结果如下：

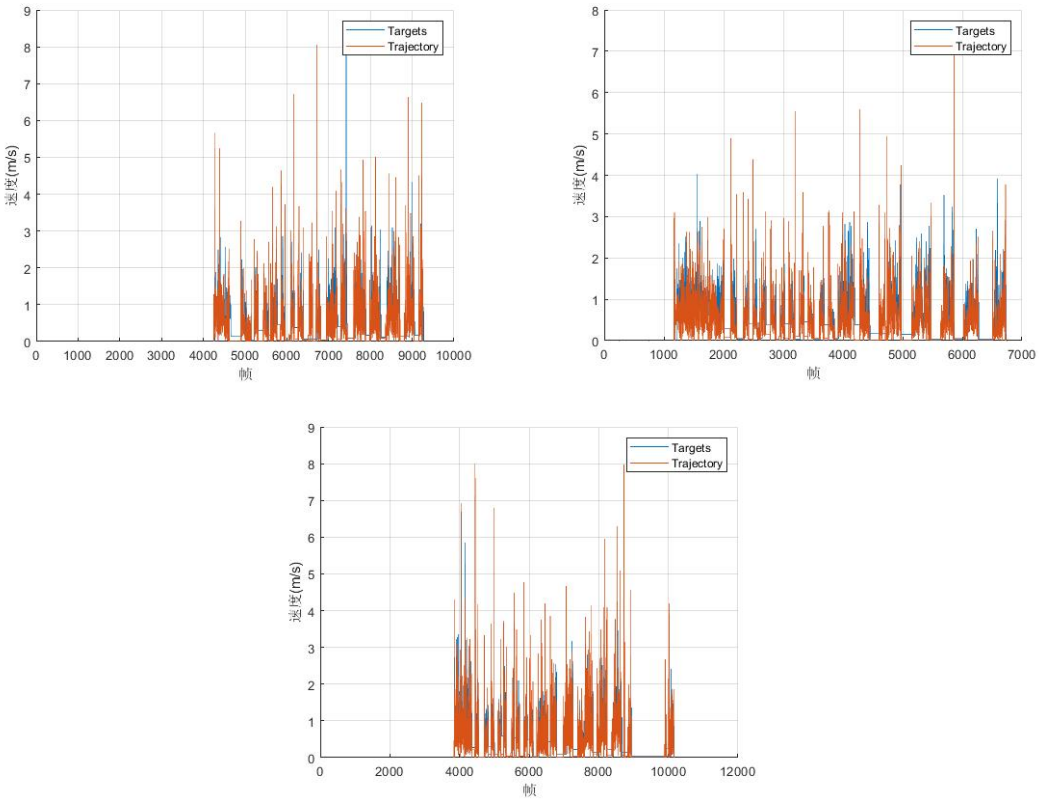

图 3 真实速度与估算速度

注：依次为 data_1c、data_2c 和 data_4z 数据的真实速度与估算速度

从实验结果可以观察到，估计速度与真实速度在数值范围上具有一致性：在真实速度为零的区域，估计速度同样趋近于零；而在真实速度非零的范围内，估计速度与真实速度的取值基本吻合。然而，估计速度曲线中出现了较多的尖峰现象，经分析，这可能是由于人体不同部位在运动过程中速度差异显著所致。

4. 实验中遇到的问题。

1. 滤除噪声点遇到的问题：

针对不同运动物体的噪声剔除，最小邻域半径（radius）的选择存在显著差异。在本实

验中，由于人体点云数据的复杂性，确定合适的 radius 值具有一定挑战性。最初，我们计划根据运动轨迹误差动态调整 radius，但这一方法显著增加了实验的复杂性和计算成本。经过综合考虑，最终将 radius 设定为 0.5 米，这一选择不仅符合人体的基本构造特征，也在实验中表现出了良好的噪声剔除效果。

2. 轨迹提取与速度估计遇到的问题：

在初始实验中，我们选择以当前帧点云数据的均值作为轨迹估计点。然而，在后续的速度估计中，发现估计速度存在较大的异常值，速度尖峰甚至达到真实速度的十几倍，这与实际运动规律明显不符。考虑到点云数据的采样间隔为 55 毫秒（约 20 帧/秒），且人体运动速度通常不会发生剧烈变化，我们优化了轨迹估计方法，采用前后各 9 帧（共 19 帧）点云数据的均值作为当前帧的轨迹估计点。这一改进显著降低了轨迹估计和速度估计的误差，使结果更加符合实际运动规律。

5. 代码文件

以下是本实验的主函数代码，子函数已在附件中提供。主函数集成了数据加载、噪声剔除、轨迹提取、速度估计以及结果可视化等功能，确保实验流程的完整性和可重复性。

	主函数 main
	% 2025 年 1 月 12 日 李厚华 空天信息创新研究院 lihouhua24@mail.ucas.ac.cn clear all; clc; close all; % 文件路径 file_path = 'D:\CODES_MATLAB\SSP\SSP_data\data_2c.jsonl'; % 读取数据 [frames, point_clouds, targets] = read_jsonl(file_path);

```
% 剔除异常值
cleaned_point_clouds = clean_data(point_clouds, targets);

% 通过临近点数量判断是否为噪声点
[filtered_point_clouds, noise_point_clouds] = filter_noise(cleaned_point_clouds);

% 提取运动轨迹
trajectory = extract_trajectory(filtered_point_clouds);

% 计算真实轨迹与估计轨迹的欧氏距离偏差
avg_distance = compute_avg_euclidean_distance(targets, trajectory);
disp(['平均欧氏距离偏差: ', num2str(avg_distance), ' 米']);

% 估计运动速度
speeds_trajectory = estimate_speed(trajectory);
speeds_targets = estimate_speed(targets);

% 画图参数设置
pause_time = 55 * 10^(-3) * 1 / 20; % 显示速度相较于原始速度加快 20 倍（55ms * 1/20）

% 绘制 3D 和 2D 轨迹图
plot_all_points(targets, trajectory); % 3D 轨迹图
plot_all_points_2d(targets, trajectory); % 2D 轨迹图

% 动态轨迹图
dynamic_point_cloud_display_fixed_axis(targets, trajectory, filtered_point_clouds, ...
    noise_point_clouds, speeds_trajectory, frames, pause_time);

% 绘制速度图
plot_speeds(speeds_targets, speeds_trajectory); % 速度对比图
```

6. 参考文献

[1] 赵卓不凡. 一文弄懂 DBSCAN 聚类算法[EB/OL](2024–06–30)[2025–01–12]. https://blog.csdn.net/sgzqc/article/details/140078475.

[2] 佚名. 机器学习 聚类篇——DBSCAN 的参数选择及其应用于离群值检测_dbscan 参数-CSDN 博客[EB/OL]([日期不详])[2025–01–12]. https://blog.csdn.net/Cyrus_May/article/details/113504879.