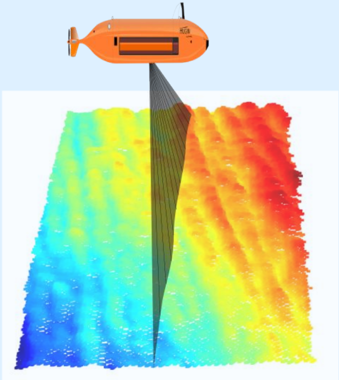
Score-Based Multibeam Point Cloud Denoising

Li Ling[1], Yiping Xie[1], Nils Bore[2], John Folkesson[1]

1 Division of Robotics, Perception and Learning (RPL), KTH Royal Institute of Technology, Sweden
2 Ocean Infinity, Sweden



Introduction

Why a learning-based multibeam denoising algorithm?

- Exponential growth in openly available MBES data [1]:
  - Cheaper sensors
  - Global initiatives to map the ocean, e.g. GEBCO Seabed 2030 [2]
- Raw MBES needs to be processed:
  - 1 - 25% outliers [1]
  - Existing data processing procedure requires data experts
- Q: Can we develop a semi-automatic MBES cleaning algorithm that is requires less manual intervention and is more repeatable and scalable?

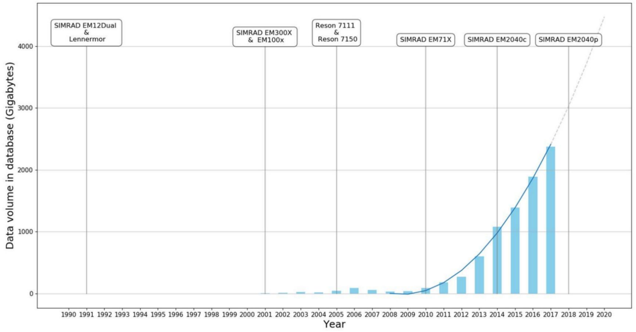
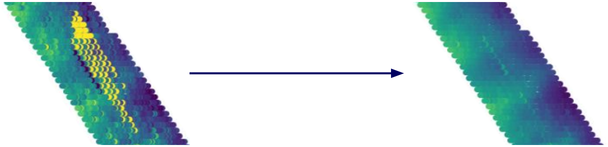


Figure 1. Volume of post-processed bathymetric data at Shom between 1991 and 2017. The last three years are missing due to the time required to integrate MBES data in the bathymetric database. [1]

[1] Le Deunf, Julian, et al. "A review of data cleaning approaches in a hydrographic framework with a focus on bathymetric multibeam echosounder datasets." Geosciences 10.7 (2020): 254.
[2] Mayer, Larry, et al. "The Nippon Foundation—GEBCO seabed 2030 project: The quest to see the world's oceans completely mapped by 2030." Geosciences 8.2 (2018): 63.

Method
Problem Formulation

- Goal: given the noisy *raw* MBES point cloud **R**, recover the *clean* correspondence **C**



$\mathbf{R} = \{\mathbf{r}_i\}_{i=1}^{N}$ $\mathbf{C} = \{\mathbf{c}_i\}_{i=1}^{N}$

- Assumption for score-based methods [3]:
  - **C** is sampled from an underlying distribution *p* (true seafloor)
  - **R** is sampled from *p* convolved with a noise distribution *n* + additional outliers *o*
  - Without the outliers, the *mode* of (p*n) = **C** -> Gradient / score of (p*n) can be used to move the points

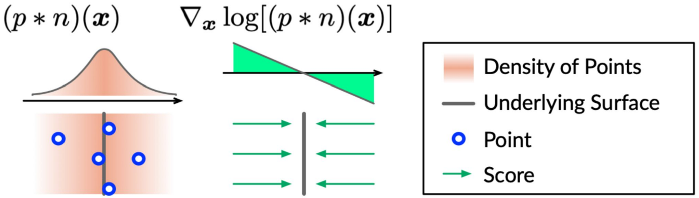


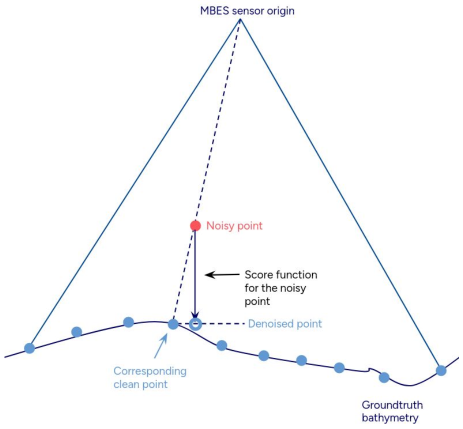
Illustration of the denoising theory [3]

[3] S. Luo and W. Hu, "Score-Based Point Cloud Denoising (Learning Gradient Fields for Point Cloud Denoising)," in 2021 IEEE/CVF International Conference on Computer Vision (ICCV), pp. 4563–4572.

Method

Problem Formulation - *score* for MBES Point Cloud

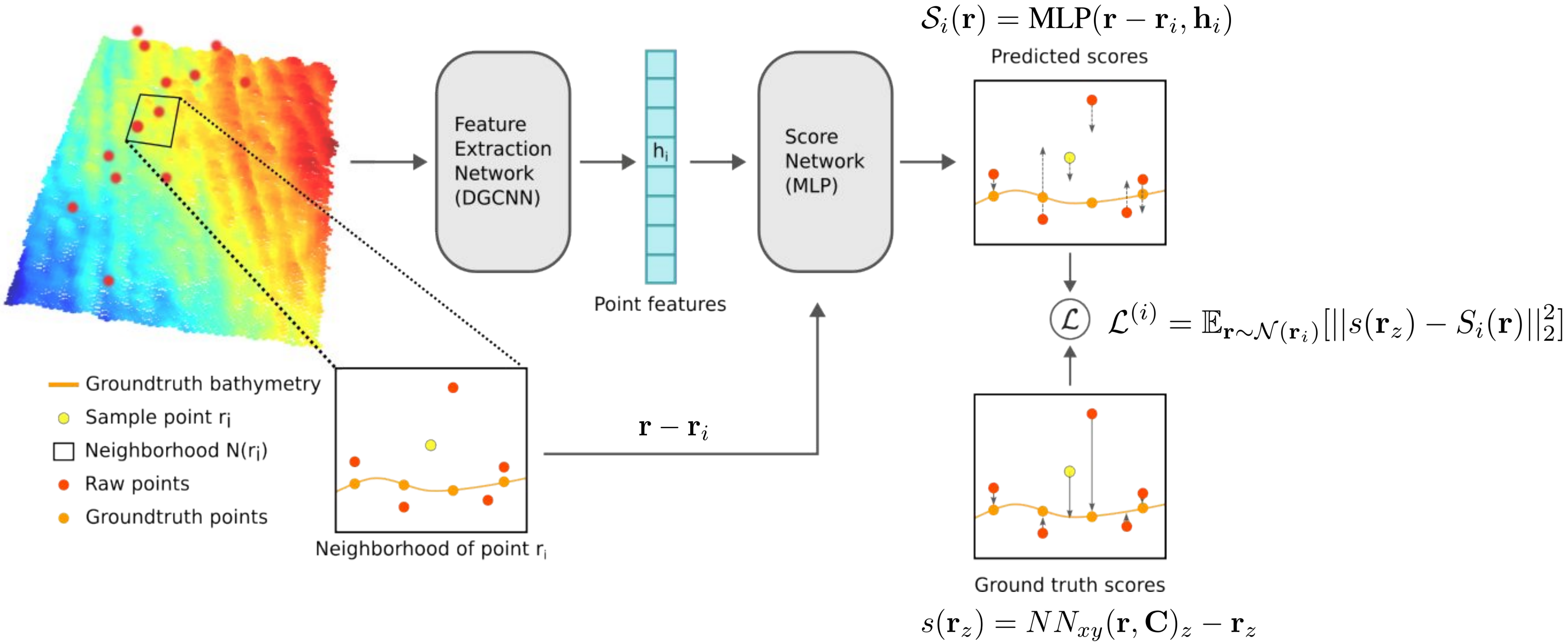
- Groundtruth score:
    - Assumptions /simplifications:
        - X positions (along-track) are fixed (i.e. each MBES ping forms a plane with the seafloor)
        - Y positions (across-track) can be moved according to MBES geometry given Z
    - Intuition: 1D scalar (z-component) from the noisy point *r* to its closest corresponding clean point *c*



$$s(\mathbf{r}_z) = NN_{xy}(\mathbf{r}, \mathbf{C})_z - \mathbf{r}_z$$

Method

Supervised Learning for the Local Score Functions

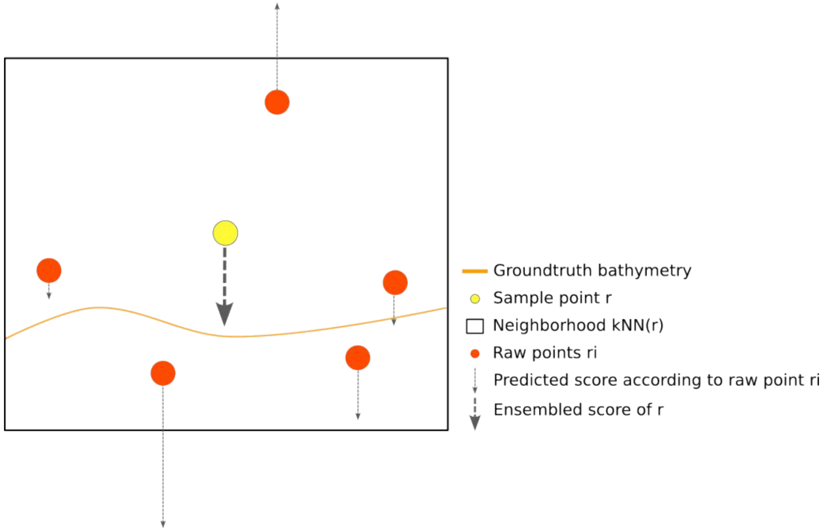


Method

Score-Based Denoising for MBES Point Clouds

- After training, the score network can predict the score of any points
- At inference time, the final score of a raw point **r**:

$$\mathcal{E}(\mathbf{r}) = \text{median}_{\mathbf{r}_i \in kNN(\mathbf{r})} S_i(\mathbf{r})$$



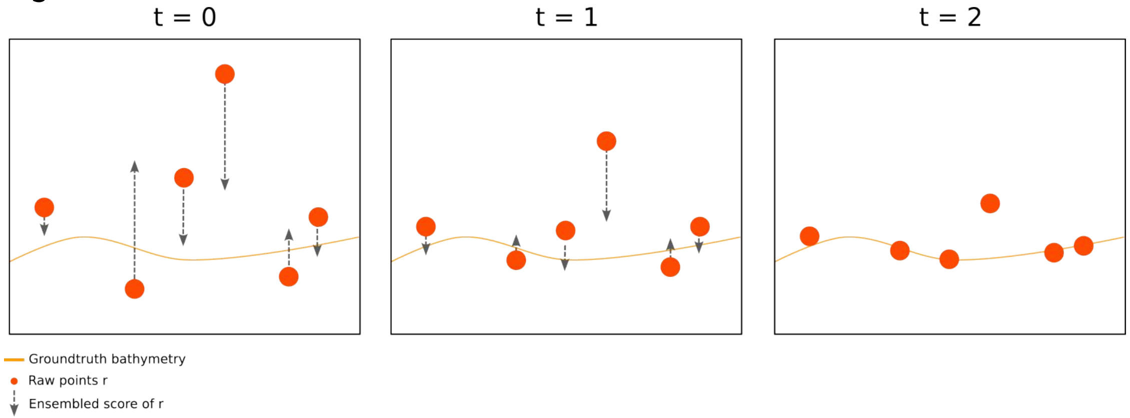
Method

Score-Based Denoising for MBES Point Clouds

- Outlier detection:
    - Interquartile range (IQR) from descriptive statistics
    - Inliers = [Q1 - i*IQR, Q3 + i*IQR]
        - Q1 = 25 percentile, Q3 = 75 percentile, IQR = Q3 - Q1, i = 5
- Denoising:
    - Iterative gradient ascend using the score

$$z_i^{(0)} = z_i$$

$$z_i^{(t)} = z_i^{(t-1)} + \alpha_i \mathcal{E}(z_i^{(t-1)})$$

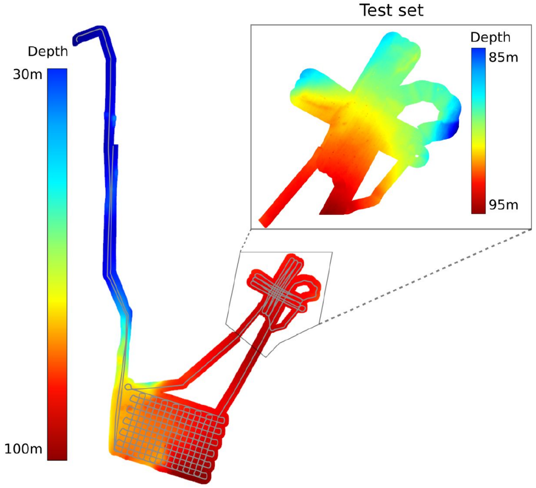


Experiments

Dataset Construction

- Dataset construction pipeline:
  a. Manual cleaning of a MBES dataset collected by Ran using Kongsberg EM2040
  b. Mesh construction using EIVA NaviModel
  c. Draping pings onto mesh to obtain groundtruth clean point set using AuvLib
  d. Dividing data into 32-ping patches for training and evaluation
- Dataset details:

Details	Specifications
Vehicle speed	2 m/s
Vehicle altitude	$\sim$ 20 m
Survey duration	$\sim$ 4 h
Sonar frequency	400 kHz
Ping rate	2.5 Hz ($\sim$ 0.4 s/ping)
Beam forming	400 beams across 120°
Total number of points	86,710,800 points (217k pings)
Total number of outliers	3,410,764 points (3.93% of all points)
Number of test points	25,518,400 points (64k pings)
Number of test outliers	1,880,800 points (7.37% of test points)

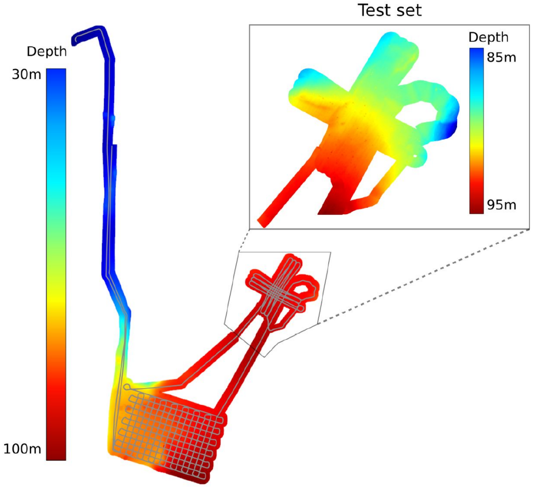


Clean bathymetry + AUV trajectory

Experiments
Dataset Construction

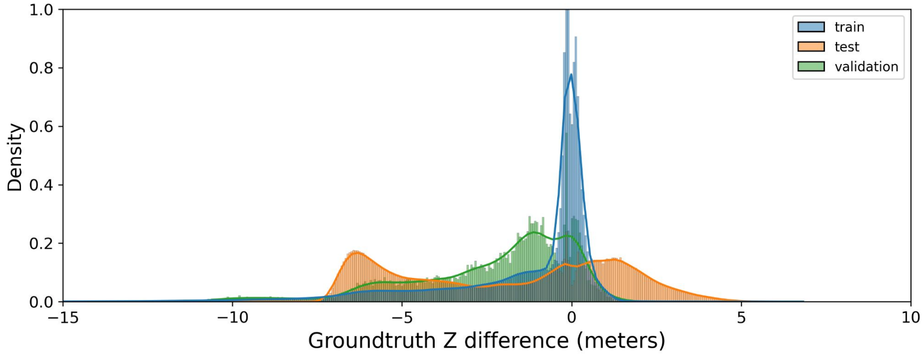
- Dataset construction pipeline:
  a. Manual cleaning of a MBES dataset collected by Ran using Kongsberg EM2040
  b. Mesh construction using EIVA NaviModel
  c. Draping pings onto mesh to obtain groundtruth clean point set using AuvLib
  d. Dividing data into 32-ping patches for training and evaluation
- Dataset details:

Details	Specifications
Vehicle speed	2 m/s
Vehicle altitude	$\sim$ 20 m
Survey duration	$\sim$ 4 h
Sonar frequency	400 kHz
Ping rate	2.5 Hz ($\sim$ 0.4 s/ping)
Beam forming	400 beams across 120°
Total number of points	86,710,800 points (217k pings)
Total number of outliers	3,410,764 points (3.93% of all points)
Number of test points	25,518,400 points (64k pings)
Number of test outliers	1,880,800 points (7.37% of test points)



Clean bathymetry + AUV trajectory

Results
Outlier Detection



- Baselines:
  - Two methods from Open3D library:
    - Statistical outlier removal
    - Radius outlier removal
  - Both baselines are tuned on the ***test set*** to ensure strong performance
- Results:

TABLE II: Outlier rejection results. The best and second best method per metric are highlighted in red and magenta, respectively.

Method	Accuracy	Precision	Recall	F1-score
Radius	98.49%	98.79%	80.61%	0.8878
Statistical	98.17%	90.76%	83.70%	0.8709
Score (64)	99.22%	99.67%	89.78%	0.9447
Score (128)	99.46%	99.72%	92.91%	0.9620
Score (256)	99.50%	99.31%	93.91%	0.9653

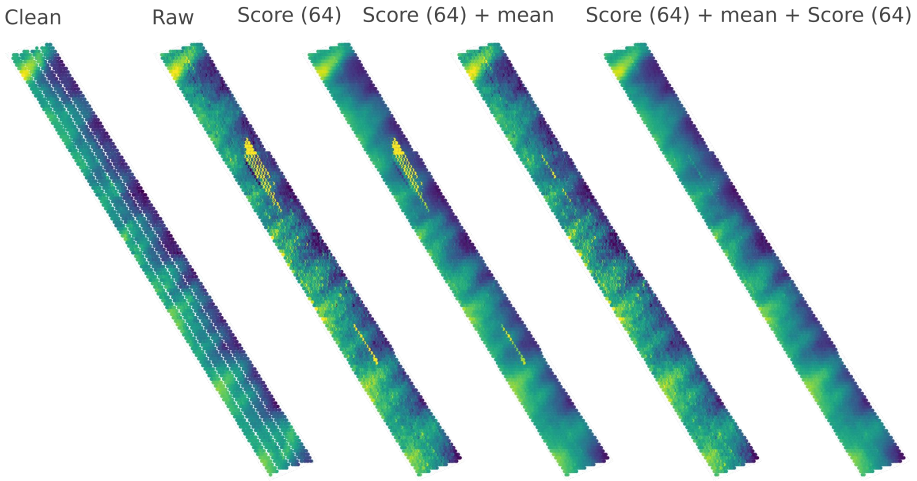
Same ScoreNet with different number of neighbors in the final score ensemble

Results
Denoising

- Baselines:
  - Radius / statistical outlier removal + 2 interpolation techniques:
    - Mean interpolation using 16 points around the identified outlier
    - Ordinary Kriging implemented in PyKrige package
- Results:

TABLE III: Denoising results. The best and second best methods per metric are highlighted in red and magenta, respectively.

	Method	CD	MAE_z	$RMSE_z$
	Raw	1.1058	0.2418	0.9136
	Radius + mean	0.07983	0.02788	0.06153
	Statistical + mean	0.08127	0.02719	0.06242
	Radius + Ordinary Kriging	0.08173	0.02838	0.06292
	Statistical + Ordinary Kriging	0.08173	0.02774	0.06394
Pure gradient ascend	Score (64)	0.2773	0.06483	0.3748
	Score (128)	0.2037	0.05478	0.2806
	Score (256)	0.1201	0.04405	0.1714
Outlier detection w/ Scores + denoising w/ mean interp.	Score (64) + mean	0.2450	0.05471	0.3368
	Score (128) + mean	0.1640	0.03994	0.2301
	Score (256) + mean	0.08348	0.02397	0.05759
Score (knn) + mean + gradient ascend denoising using Scores	Score (64) + mean + Score (64)	0.08790	0.02327	0.1043
	Score (128) + mean + Score (128)	0.08416	0.02208	0.09158
	Score (256) + mean + Score (256)	0.07907	0.02049	0.07416



Results
Denoising

- Baselines:
  - Radius / statistical outlier removal + 2 interpolation techniques:
    - Mean interpolation using 16 points around the identified outlier
    - Ordinary Kriging implemented in PyKrige package
- Results:

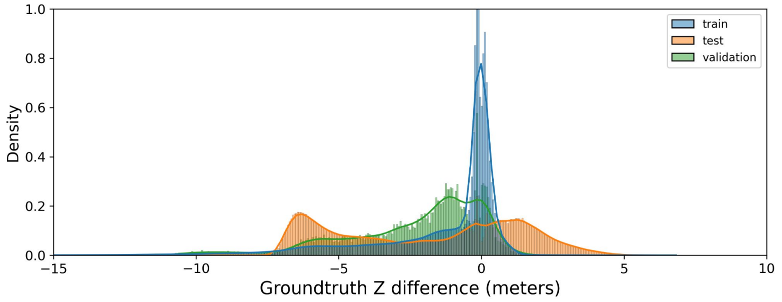
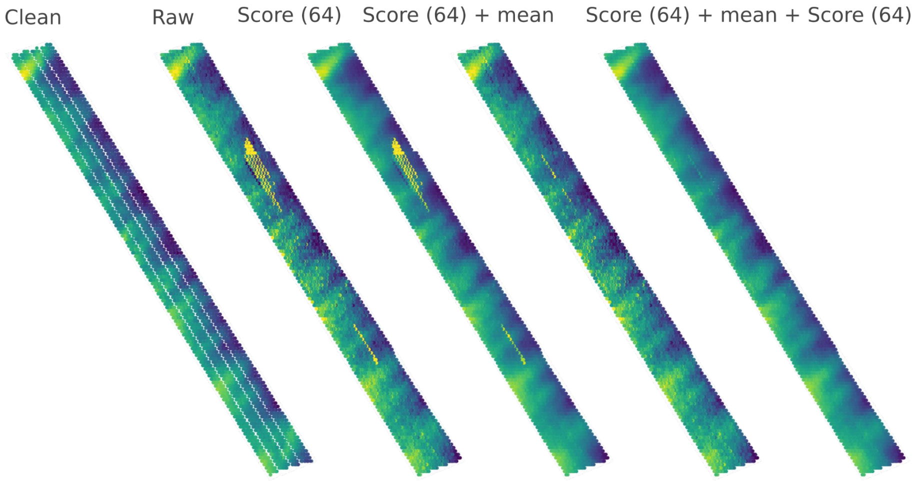


TABLE III: Denoising results. The best and second best methods per metric are highlighted in red and magenta, respectively.

	Method	CD	MAE_z	$RMSE_z$
	Raw	1.1058	0.2418	0.9136
	Radius + mean	0.07983	0.02788	0.06153
	Statistical + mean	0.08127	0.02719	0.06242
	Radius + Ordinary Kriging	0.08173	0.02838	0.06292
	Statistical + Ordinary Kriging	0.08173	0.02774	0.06394
Pure gradient ascend	Score (64)	0.2773	0.06483	0.3748
	Score (128)	0.2037	0.05478	0.2806
	Score (256)	0.1201	0.04405	0.1714
Outlier detection w/ Scores + denoising w/ mean interp.	Score (64) + mean	0.2450	0.05471	0.3368
	Score (128) + mean	0.1640	0.03994	0.2301
	Score (256) + mean	0.08348	0.02397	0.05759
Score (knn) + mean + gradient ascend denoising using Scores	Score (64) + mean + Score (64)	0.08790	0.02327	0.1043
	Score (128) + mean + Score (128)	0.08416	0.02208	0.09158
	Score (256) + mean + Score (256)	0.07907	0.02049	0.07416



Conclusions

- We adapt a score-based point cloud denoising to MBES survey data
- We propose a training data generation pipeline that can be readily integrated into existing MBES data processing workflow
- For *outlier detection*, the score-based method outperforms all baselines
- For *denoising,* we combine score-based denoising and mean interpolation to handle extreme outliers

Code: