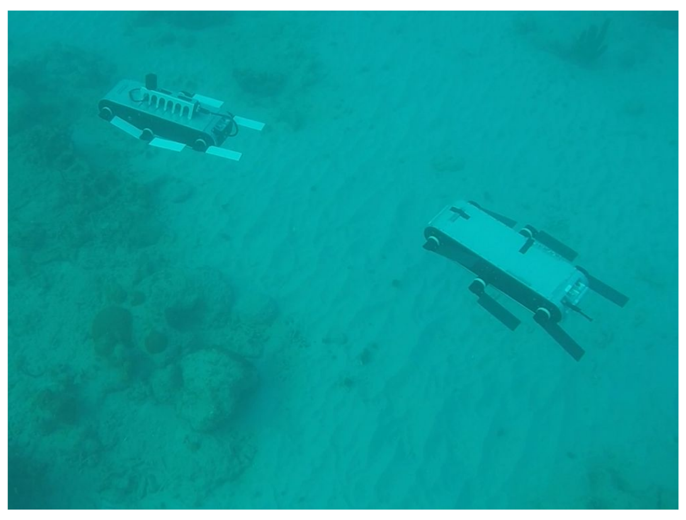
Underwater Communication Using Full-Body Gestures and Optimal Variable-Length Prefix Codes

Karim Koreitem(*), Jimmy Li(*), Ian Karp(*), Travis Manderson(*), and Gregory Dudek (*)(^)
** School of Computer Science, Centre for Intelligent Machines, McGill University*
^ Samsung AI Center, Montreal

Problem



Objective:
- Communication for multi-robot system in radio-denied environments (i.e. underwater)
- Use whole-body gestures to convey messages

Problem:
- Encode messages based on robot pose configurations (LOW BATTERY, U-TURN, START MAPPING)
- Gestures should be unambiguous and reliably detected with RGB sensor

Motivation

Multi-robot systems are less prone to a single point of failure.

Gesturing is a suitable communication method in many of these robotic settings:
- RGB cameras are a ubiquitous sensor
- Gestures are easily interpretable and often used by humans
  - Aircraft marshalling
  - Hand gesturing by scuba divers
  - Cyclists signalling their trajectory

Evaluation

We validate our system on the highly maneuverable **Aqua** family of amphibious hexapod robots.

Simulated gestures: 50 examples of 8 different messages executed in a simulated underwater environment

Real pool gestures: 10 examples of 6 different messages executed in a real pool

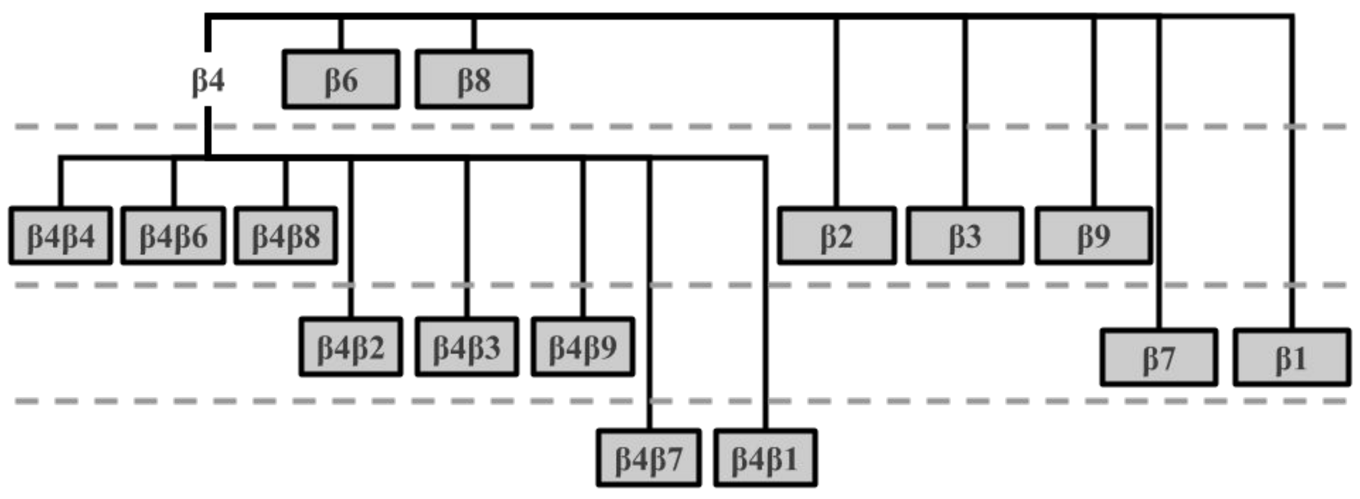
Simulated swimming trajectories with gesturing: 10 simulated trajectories with 10 different messages executed 5 times at random over 2min of navigation

Method

1. **Optimal prefix-free encoding of poses**

We encode messages as a sequence of binned orientations (**codebits β_i**) using a generalization of Huffman coding.

An example optimal prefix-free code tree encoding 15 messages using 8 codebits:



Leaves of the tree (**in gray**) represent the final codebit sequences (**codewords**) that make up the code.

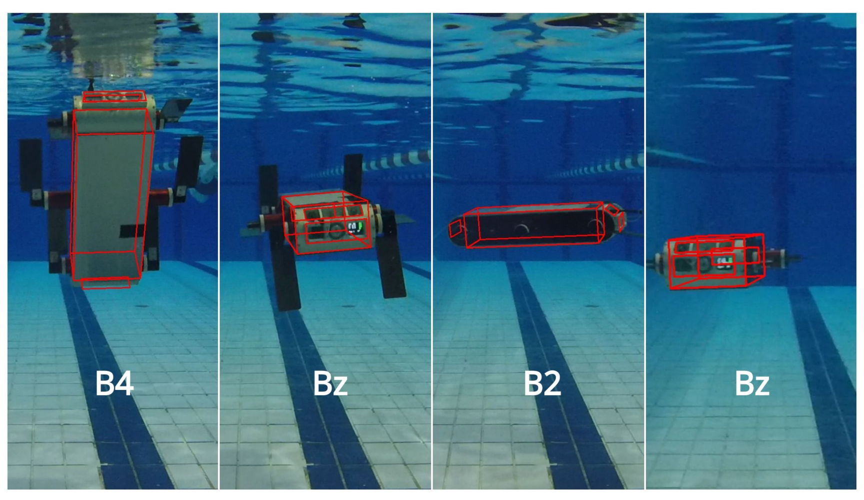
Each codebit has an associated **transmission cost T** (depth of the tree) that combines:
- **$p(\beta_i)$:** probability of a codebit in regular operation
- **$e(\beta_i)$:** normalized mean error of the orientation regressor
- **$d(\beta_i)$:** application-specific value which can represent the time it takes to execute a codebit, or other engineering restrictions

$$T(\beta_i) = p(\beta_i) \cdot e(\beta_i) \cdot d(\beta_i)$$

E.g: yaw may be more expensive than roll because it is a high probability codebit during normal operation.

2. **Visual decoder**

CNN-based orientation regressor:
- Trained on synthetic data
- Generalizes to real images



Message decoder:
- Bin the orientation prediction on every frame
- Check for codebit sequences that match our codewords

Results

	$\beta_4\beta_4$	$\beta_4\beta_2$	$\beta_4\beta_1$	$\beta_4\beta_7$	β_2	β_1	β_3	β_7	**FNs**
$\beta_4\beta_4$	0.94	0.	0.	0.	0.	0.	0.	0.	0.06
$\beta_4\beta_2$	0.04	0.82	0.	0.	0.08	0.	0.	0.	0.06
$\beta_4\beta_1$	0.02	0.06	0.84	0.	0.	0.04	0.	0.	0.04
$\beta_4\beta_7$	0.	0.	0.	0.98	0.	0.	0.	0.02	0.
β_2	0.	0.	0.	0.	0.94	0.	0.	0.	0.06
β_1	0.	0.	0.	0.	0.04	0.86	0.	0.	0.10
β_3	0.	0.	0.	0.	0.	0.	1.	0.	0.
β_7	0.	0.	0.	0.	0.	0.	0.	0.90	0.10
Precision	0.94	0.93	1.00	1.00	0.89	0.96	1.00	0.98	

Confusion matrix for codewords executed in simulation:
- Mean precision: 0.96
- Mean recall: 0.91
- Successfully decodes messages in simulation

	$\beta_4\beta_4$	$\beta_4\beta_2$	β_2	β_1	β_6	β_8	**FNs**
$\beta_4\beta_4$	0.67	0.	0.	0.	0.	0.17	0.17
$\beta_4\beta_2$	0.	0.67	0.17	0.	0.	0.	0.17
β_2	0.	0.	0.73	0.	0.	0.	0.27
β_1	0.	0.	0.38	0.25	0.	0.	0.38
β_6	0.	0.	0.	0.	0.83	0.	0.17
β_8	0.	0.	0.	0.	0.	0.82	0.18
Precision	1.00	1.00	0.67	1.00	1.00	0.95	

Confusion matrix for codewords executed in the pool:
- Codebit β_1 (yaw: -60°, pitch: -60°) was mistaken as β_2 (yaw: -60°, pitch: 0°) due to under-pitching by the controller
- A solution is to use codebits that are more spread out in the orientation space

Trajectory	Codeword	Counts	Precision	Recall
1	$\beta_4\beta_4$	5	0.83	1.
2	$\beta_4\beta_2$	5	0.83	1.
3	$\beta_4\beta_1$	5	0.67	0.80
4	$\beta_4\beta_7$	5	0.71	1.
5	β_2	5	0.83	1.
6	β_1	5	0.71	1.
7	β_8	5	0.67	0.80
8	β_3	5	1.	1.
9	β_6	5	0.83	1.
10	β_8	5	0.63	1.
		Mean	**0.77**	**0.96**

Precision/Recall of the decoded messages executed during synthetic trajectories:
- Common false negatives are codebits β_2 and β_8 which represent basic left/right yaw orientations
- These codebits have a high probability of occurrence in regular deployment

Mobile Robotics Lab (MRL): http://www.cim.mcgill.ca/~mrl/
Paper: https://ryteim.github.io/research/