BADGR: An Autonomous Self-Supervised Learning-Based Navigation System

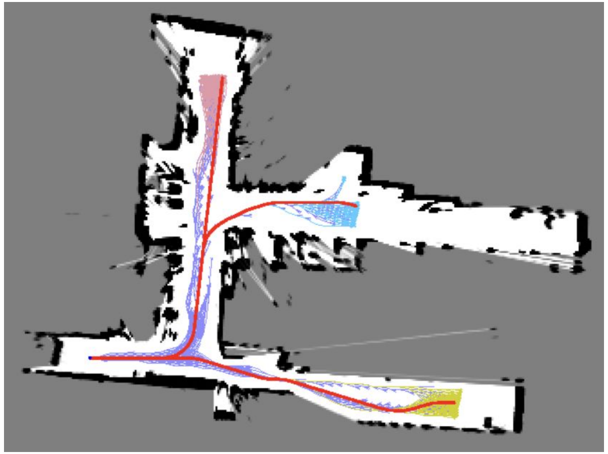
Gregory Kahn, Pieter Abbeel, Sergey Levine
Berkeley AI Research (BAIR), University of California, Berkeley

Robotics and Automation Letters, 2021

Motivation

- Navigation for mobile robots is often regarded as primarily a **geometric problem.**

  (Construct map → Plan path on the map → Track the planned path)

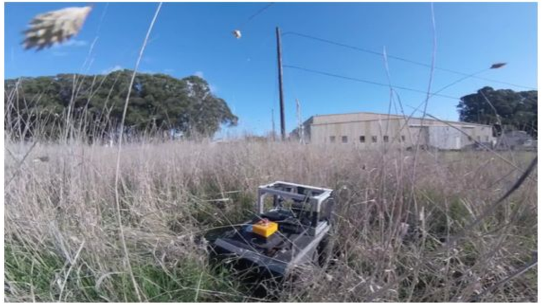


Motivation

- Navigation for mobile robots is often regarded as primarily a **geometric problem.**

  (Construct map → Plan path on the map → Track the planned path)

- However, these methods **could not** handle terrain properties and physical attributes of the environment for deciding the traversability. (ex) tall grass)


Motivation

- Navigation for mobile robots is often regarded as primarily a **geometric problem.**

  (Construct map → Plan path on the map → Track the planned path)

- However, these methods **could not** handle terrain properties and physical attributes of the environment for deciding the traversability. (ex) tall grass)

Let’s learn the traversability from experience.

Contribution

The paper propose **BADGR**,

- an end-to-end learning-based mobile robot navigation system
- that can be trained entirely with self- supervised,
- off-policy data gathered in real-world environments,
- without any simulation or human supervision (auto labeling),
- and can improve as it gathers more data.
- **Model based RL** for decision making.
- Perform point goal navigation task in an unknown environment. **(no prior knowledge of the environment)**


BADGR: An Autonomous Self-Supervised Learning-Based Navigation System
Gregory Kahn, Pieter Abbeel, Sergey Levine
BAIR
BERKELEY ARTIFICIAL INTELLIGENCE RESEARCH

Method

- Mobile robot platform
- Data collection
- Self-supervised data labelling
- Predictive model
- Planning

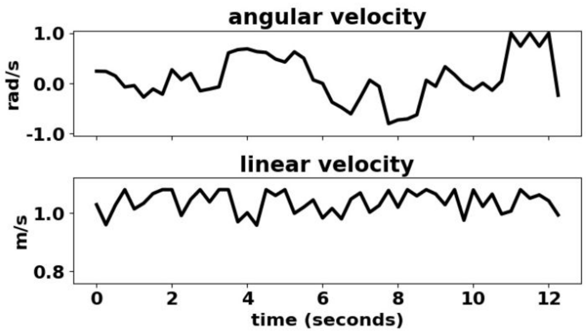
Method : Mobile robot platform

- Wheeled mobile robot
- Sensor: Camera (for BADGR), 2D Lidar (for Baseline), IMU, GPS
- Onboard computing: NVIDIA Jetson TX2

linear acceleration, angular velocity

Method : Data collection

- Gather large amounts of diverse data with **minimal human intervention**.
- Used time-correlated random walk control policy for data gathering. (off-policy)
- Detect collision using Lidar and IMU. If it is dangerous, reset the robot to safe coordinate.



Method : Self-supervised data labelling

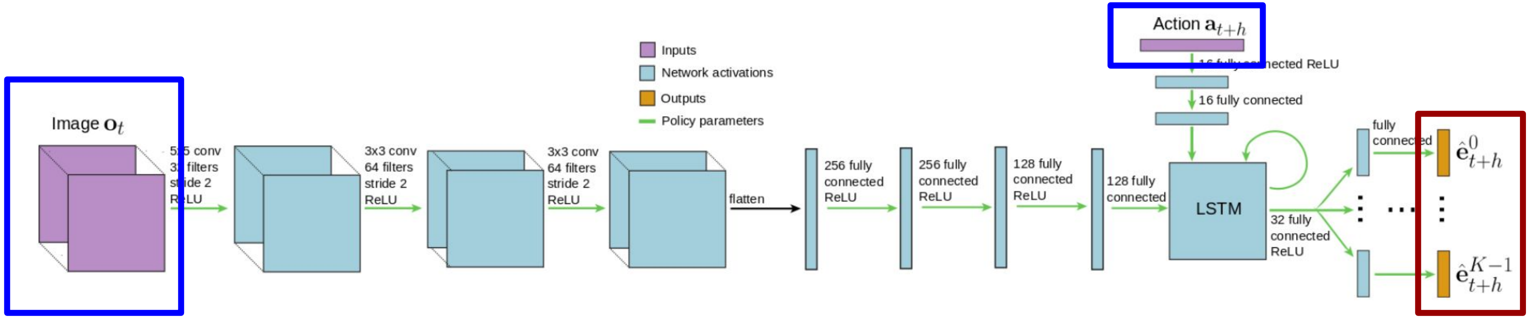
- Calculate labels for specific navigational events, which include collision, bumpiness, position.
- **Collision:** *Lidar < threshold* || *IMU magnitude sudden drop*
- **Bumpiness:** |*IMU angular velocity*| > *threshold*
- **Position:** *Wheel odometry* + *IMU*

- Mobile robot platform
- Data collection
- Self-supervised data labelling
- **Predictive model**
- Planning

Method : Predictive model

- Predict H future steps, conditioned on current observation and H intended future command sequences.

Command sequences



Current observation

Collision
Bumpiness
Position

Method : Predictive model

- Predict *H* future steps, conditioned on current observation and *H* intended future command sequences.

Algorithm 1 Training BADGR

initialize dataset $\mathcal{D} \leftarrow \emptyset$

Collect data

while not done collecting data **do**
    get current observation $\mathbf{o}_t$ from sensors
    get action $\mathbf{a}_t$ from data collection policy
    add $(\mathbf{o}_t, \mathbf{a}_t)$ to $\mathcal{D}$
    execute $\mathbf{a}_t$
    if in collision **then**
        execute reset maneuver
    end if
end while

Self-labeling

for each $(\mathbf{o}_t, \mathbf{a}_t) \in \mathcal{D}$ **do**
    calculate event labels $\mathbf{e}_t^{0:K}$ using self-supervision
    add $\mathbf{e}_t^{0:K}$ to $\mathcal{D}$
end for

Train

use $\mathcal{D}$ to train predictive model f_θ by minimizing Eqn. 1

Method : Planning

- Gradient free optimizer + model predictive control
- Use task specific rewards which are functions of the outputs of learned predictive model

Prob.

$$\mathbf{a}^*_{t:t+H} = \arg\max_{\mathbf{a}_{t:t+H}} R(f_\theta(\mathbf{o}_t, \mathbf{a}_{t:t+H})),$$

Sol.

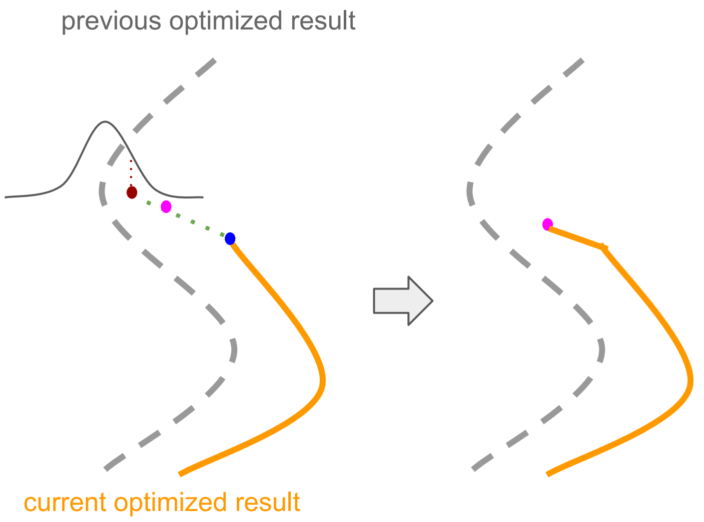
$$\epsilon^n_h \sim \mathcal{N}(0, \sigma \cdot \mathbf{I}) \; \forall n \in \{0...N-1\}, h \in \{0...H-1\}$$

$$\tilde{\mathbf{a}}^n_h = \beta \cdot \underbrace{(\hat{\mathbf{a}}_{h+1} + \epsilon^n_h)}_{\text{previous optimized distribution}} + (1-\beta) \cdot \underbrace{\tilde{\mathbf{a}}^n_{h-1}}_{\text{time correlation}} \text{ where } \tilde{\mathbf{a}}_{h<0} = 0$$

$$\tilde{R}^n = R(f_\theta(\mathbf{o}_t, \tilde{\mathbf{a}}^n_{0:H}))$$

$$\hat{\mathbf{a}}_{0:H} = \frac{\sum_{n=0}^{N} \exp(\gamma \cdot R^n) \cdot \tilde{\mathbf{a}}^n_{0:H}}{\sum_{n'=0}^{N} \exp(\gamma \cdot R^{n'})},$$

(Model based RL)



Method : Planning

- Gradient free optimizer + model predictive control
- Use task specific rewards which are functions of the outputs of learned predictive model

Prob.

$$\mathbf{a}^*_{t:t+H} = \arg\max_{\mathbf{a}_{t:t+H}} R(f_\theta(\mathbf{o}_t, \mathbf{a}_{t:t+H})),$$

Sol.

$$\epsilon^n_h \sim \mathcal{N}(0, \sigma \cdot \mathbf{I}) \ \forall n \in \{0 \ldots N-1\}, h \in \{0 \ldots H-1\}$$

$$\tilde{\mathbf{a}}^n_h = \beta \cdot \underbrace{(\hat{\mathbf{a}}_{h+1} + \epsilon^n_h)}_{\text{previous optimized distribution}} + (1-\beta) \cdot \underbrace{\tilde{\mathbf{a}}^n_{h-1}}_{\text{time correlation}} \text{ where } \tilde{\mathbf{a}}_{h<0} = 0$$

$$\tilde{R}^n = R(f_\theta(\mathbf{o}_t, \tilde{\mathbf{a}}^n_{0:H}))$$

$$\hat{\mathbf{a}}_{0:H} = \frac{\sum_{n=0}^{N} \exp(\gamma \cdot R^n) \cdot \tilde{\mathbf{a}}^n_{0:H}}{\sum_{n'=0}^{N} \exp(\gamma \cdot R^{n'})},$$

Algorithm 2 Deploying BADGR

input: trained predictive model f_θ, reward function R
while task is not complete **do**
  get current observation $\mathbf{o}_t$ from sensors
  solve Eqn. 2 using $f_\theta, \mathbf{o}_t$, and R
  to get the planned action sequence $\mathbf{a}^*_{t:t+H}$
  $\boxed{\text{execute the first action } \mathbf{a}^*_t}$
end while

Experiment

- Urban environment
- Off-road environment (e.g. tall grass)
- Self-improvement
- Generalization

Reward for “Point goal navigation”

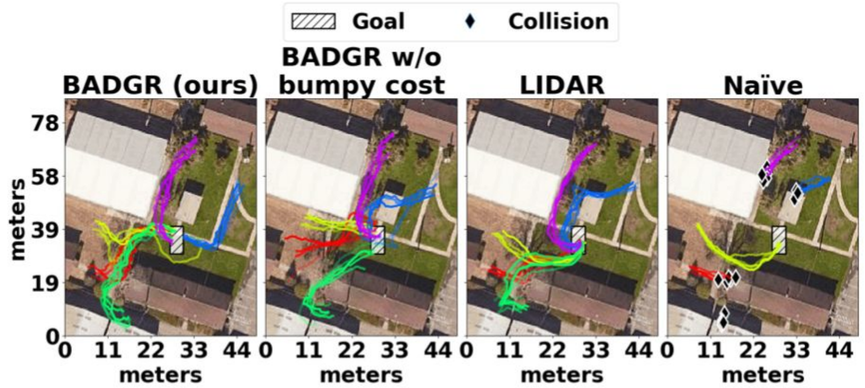
$$R^{\mathrm{COLL}}(\hat{\mathbf{e}}_{t'}^{0:K}) = \hat{\mathbf{e}}_{t'}^{\mathrm{COLL}}$$ Don’t collide

$$R^{\mathrm{POS}}(\hat{\mathbf{e}}_{t'}^{0:K}) = (1-\hat{\mathbf{e}}_{t'}^{coll}) \cdot \frac{1}{\pi} \angle(\hat{\mathbf{e}}_{t'}^{\mathrm{POS}}, \mathbf{p}^{\mathrm{GOAL}}) + \hat{\mathbf{e}}_{t'}^{coll}$$ Move to the goal direction

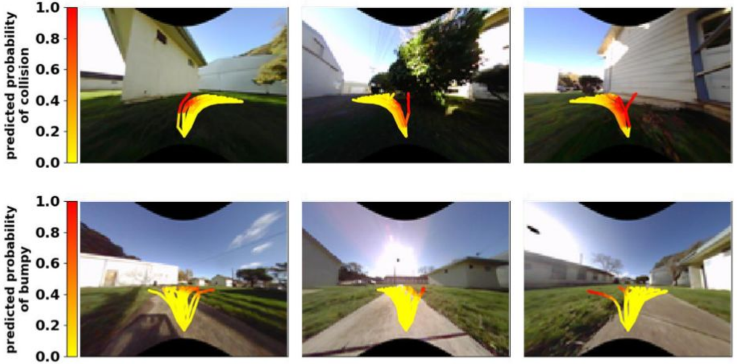
$$R^{\mathrm{BUM}}(\hat{\mathbf{e}}_{t'}^{0:K}) = (1-\hat{\mathbf{e}}_{t'}^{coll}) \cdot \hat{\mathbf{e}}_{t'}^{\mathrm{BUM}} + \hat{\mathbf{e}}_{t'}^{coll},$$ Don’t run on bumpy terrain

- **Urban environment**
- Off-road environment (e.g. tall grass)
- Self-improvement
- Generalization

Experiment : Urban environment

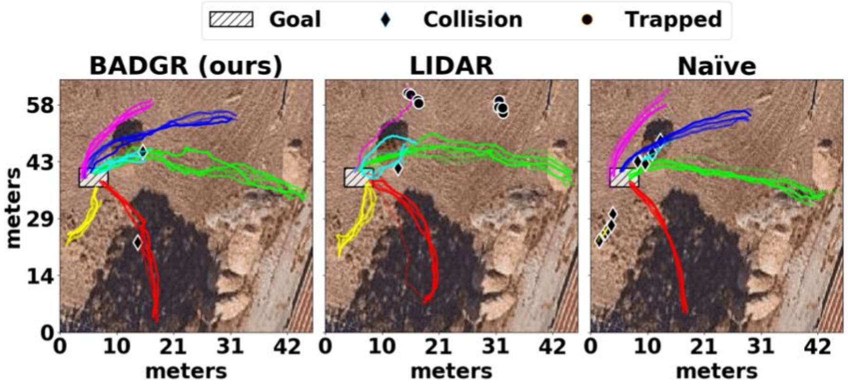


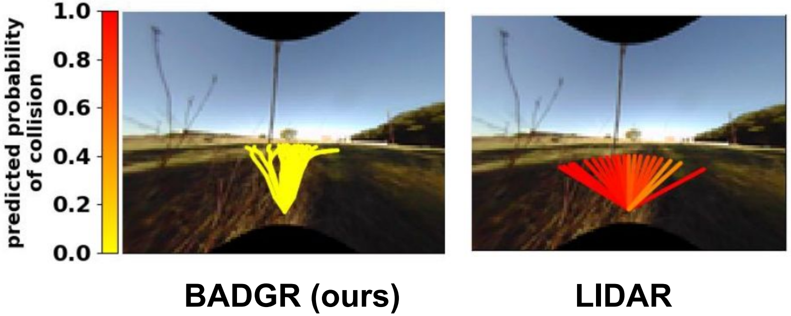
Method	Successfully Reached Goal	Avg. Bumpiness
BADGR (ours)	**25/25 (100%)**	**8.7 ± 4.4**
BADGR w/o bumpy cost	25/25 (100%)	15.0 ± 3.4
LIDAR	25/25 (100%)	13.3 ± 2.9
Naïve	5/25 (20%)	N/A



- Urban environment
- **Off-road environment (e.g. tall grass)**
- Self-improvement
- Generalization

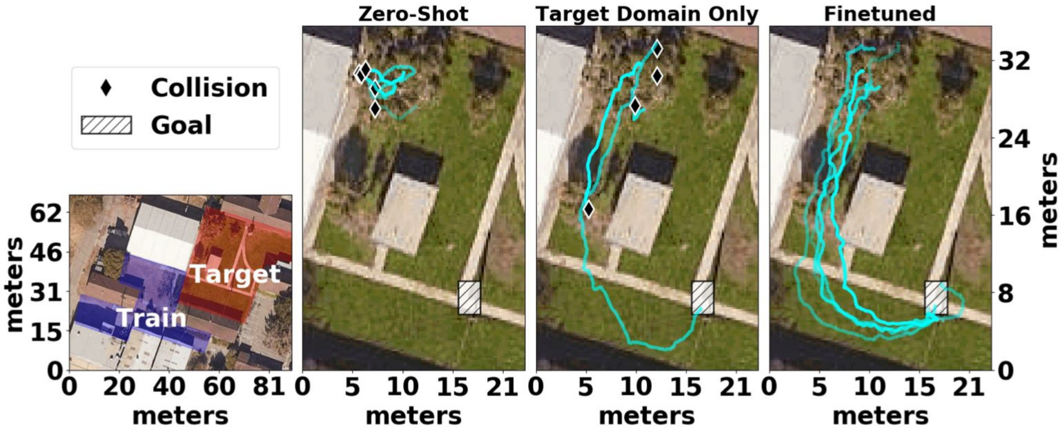
Experiment : Off-road environment





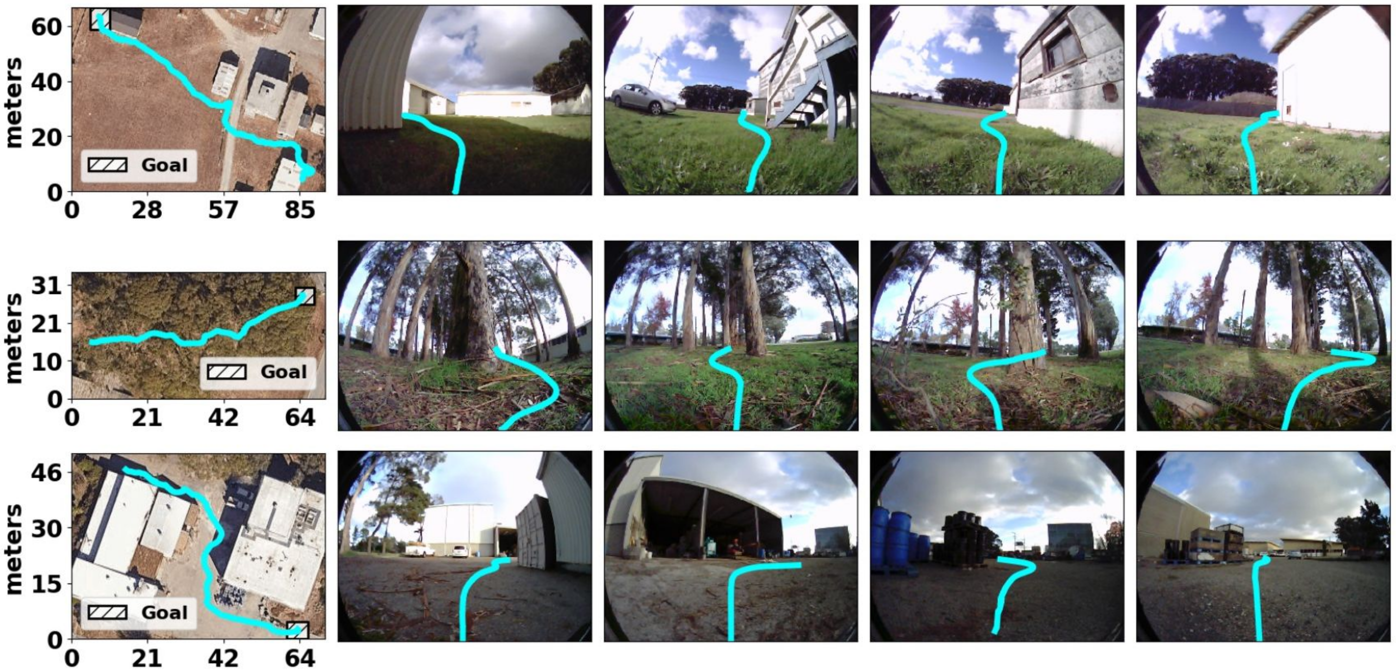
- Urban environment
- Off-road environment (e.g. tall grass)
- **Self-improvement**
- Generalization

Experiment : Self-improvement



- Urban environment
- Off-road environment (e.g. tall grass)
- Self-improvement
- **Generalization**

Experiment : Generalization



Limitations (Personal thought)

- Generalization in various environments is left as question.
- Directly running robots in real world to collect data is inefficient and not good for robots.
- Difficult to be robust for various goal positions. (Typical problem for autonomous navigation in unknown environments)
- Wobbling control outputs requires post-processing.

However, it is very simple
and works well in an unknown environment.

Project page

https://sites.google.com/view/badgr

Q & A