Photo: Anybotics

Article presentation: ANYmal parkour

Learning Agile Navigation for Legged Robots

Article by : David Hoeller
Nikita Rudin
Dhionis Sako
Marco Hutter

Short summary

Parkour

https://www.youtube.com/watch?v=VsAe02lPFeg

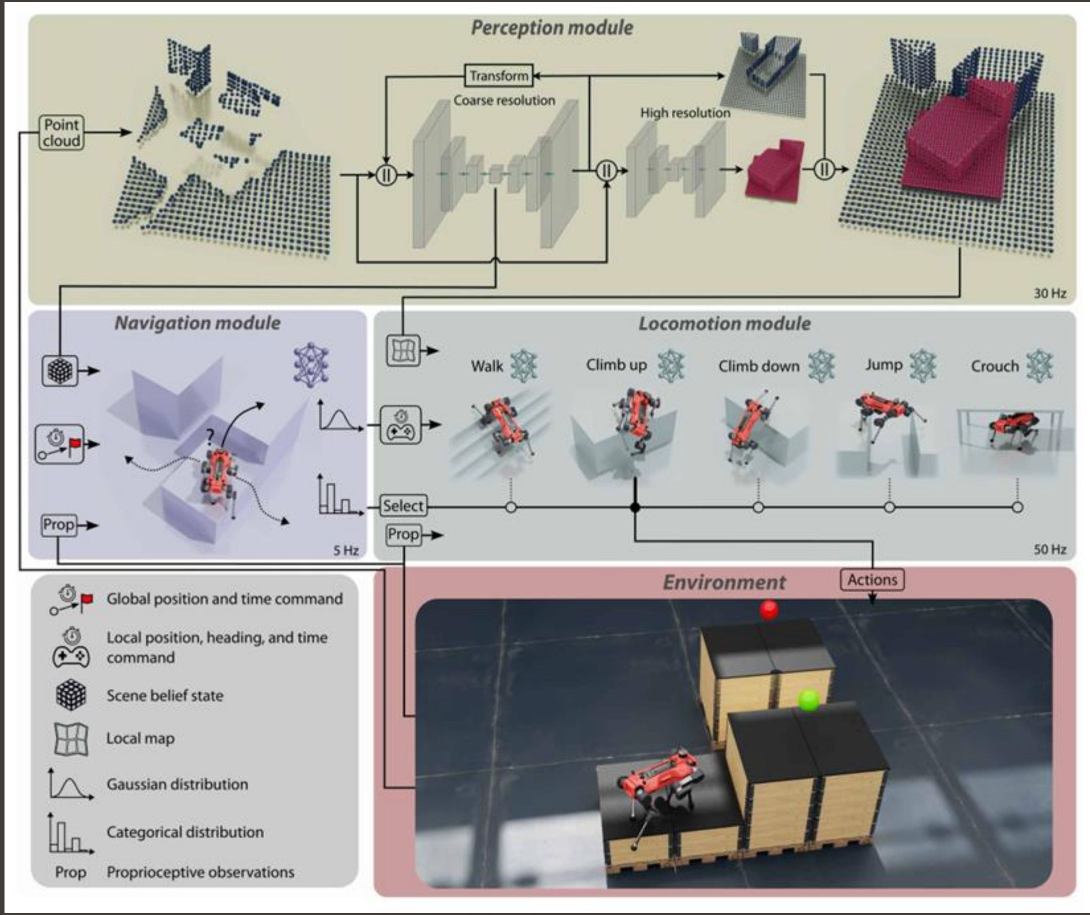
ANYmal

https://www.youtube.com/watch?v=PjWvf90l4cg

Key aspects


Reinforcement learning
360° Velodyne Puck LiDAR
Walking
Climbing up
Climbing down
Crouching
Jumping
12 series elastic actuators
6 Intel RealSense depth cameras
Up to 2 m/s
ANYmal D
55kg
Position-based

Pipeline Overview



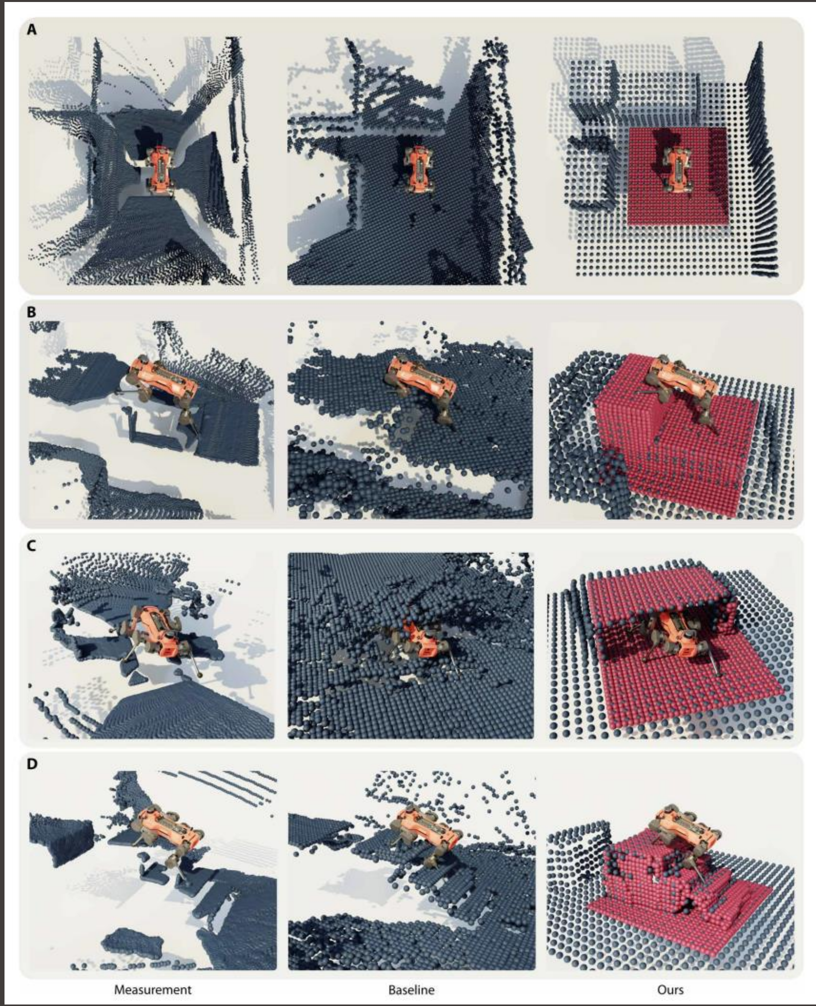
Perception Module

Input:

- Data from the LiDAR and depth camera.

Role:

- Interpret data from the sensors.
- Produce an elevation map and a two-resolutions 3D mapping of the surrounding of the robot.
- Operate at 30Hz.



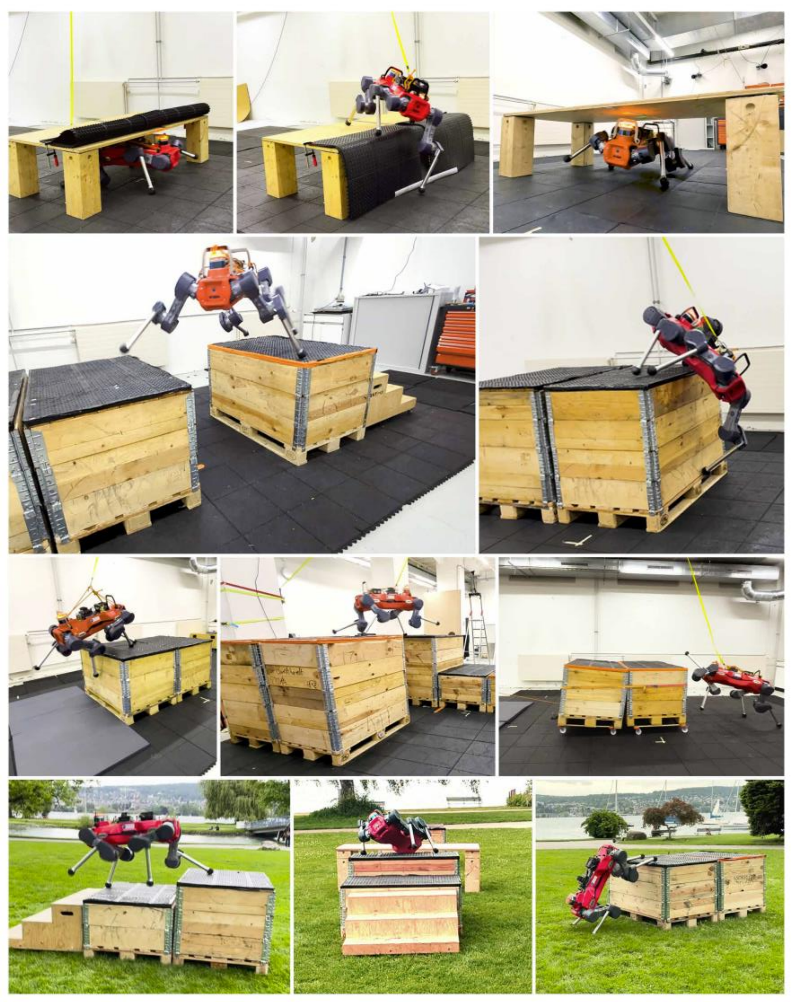
Locomotion Module

Input:

- The map of elevation from the perception module.

Role:

- Provide the robot with a set of policies trained for a specific locomotion skill (walking, climbing, crouching, jumping).
- Train skills using reinforcement learning.



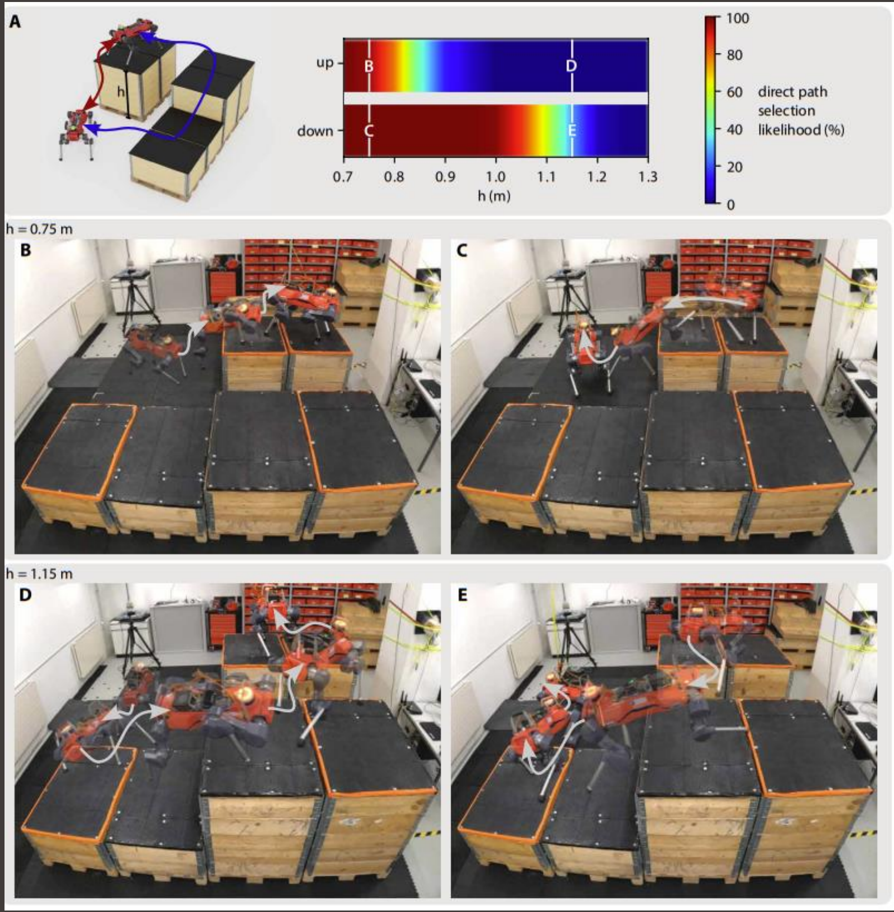
Navigation Module

Input:

- Receive the latent tensor from the perception module.
- The relative position of the goal to reach
- The time to do the task

Role:

- Uses reinforcement learning to identify a safe and efficient path through the environment.
- Guide ANYmal with a local position, heading and timer command.



Observed Results – Locomotion Module

A. Jumping
   - For jumps the robot favors a diagonal orientation and uses 3 legs to launch.

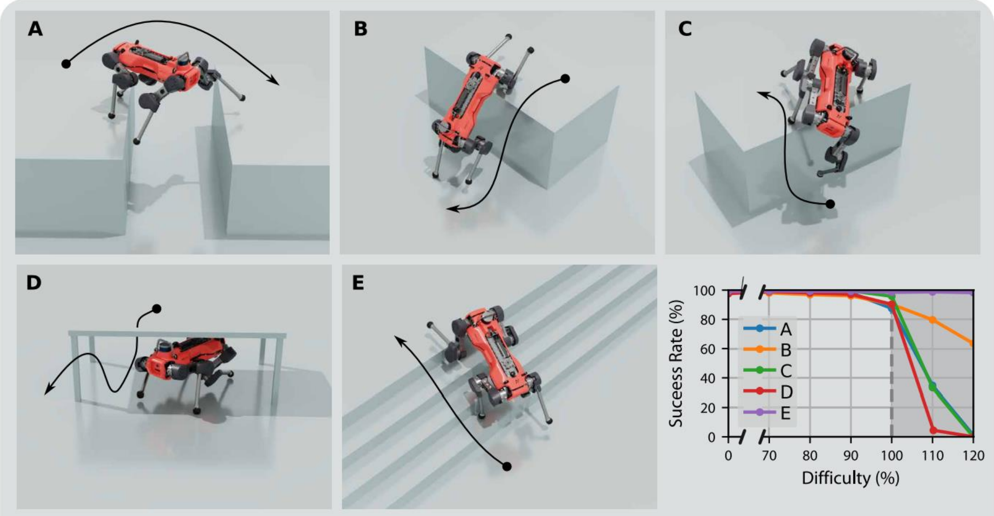
B. Climbing down
   - The robot keeps its knees on the structure as long as possible.

C. Climbing up
   - The robot places one leg on top of the obstacle and brings the rest up while using the vertical surface for balance.

D. Crouching
   - The robot adapts its gait using its hip joints to reduce its height.

E. Walking
   - The policy used is similar to other works and adapts well to many situations.

F. Success Rates
   - Represent how well skills scale when used outside intended difficulty.



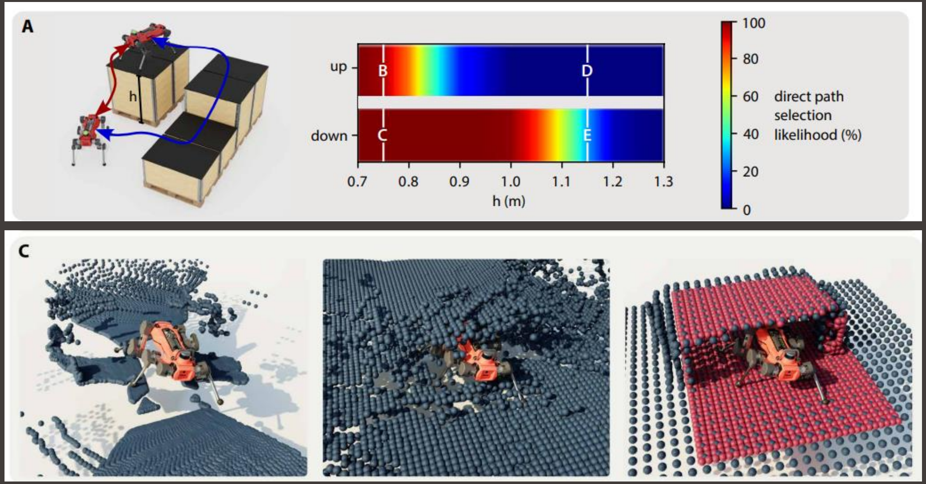
	A	B	C	D	E		
Parameter	gap size (m)	box height (m)	box height (m)	table height (m)	slope (°)	step height (m)	obstacle height (m)
Difficulty Range	[0.1, 1.2]	[0.1, 1.0]	[0.1, 1.0]	[0.8, 0.45]	[0, 45]	[0.05, 0.2]	[0.1, 0.4]

Fig. 4. Training scenarios of the locomotion skills with the resulting behaviors. (**A**) Jumping. (**B**) Climbing down. (**C**) Climbing up. (**D**) Crouching. (**E**) Walking. (**F**) The success rate of each skill for obstacles of varying difficulty. (**G**) Ranges of parameters used during training [0 to 100% in (F)].

Observed Results – Navigation and Perception

A. Navigation module
- The navigation module is capable of choosing an appropriate path based on the capabilities of different locomotion skills.
- Notable example: jump skill for turning as most efficient.
- Navigation module is more effective than hard coded trajectory.
- Appropriate skills are generally chosen for given situation.

B. Perception module
- Allows robot to perceive shapes such as a table with much higher accuracy compared to basic elevation map.



Limitations

There are a few remaining limitations with the approach presented in the article. These include:

- Untested scalability to more complex environments
- Time consuming training due to large number of models (and interdependence)
- Long convergence time due to complexity of the task the robot must complete.

Citations

The article has already been cited around 80 times despite its rather recent publication. These citations are however generally mentions to the article as examples of what has been achieved rather than mentions of it being used as inspiration or critiques. Some example citations are given below:

- X. Cheng, K. Shi, A. Agarwal and D. Pathak, "Extreme Parkour with Legged Robots," 2024 IEEE International Conference on Robotics and Automation (ICRA), Yokohama, Japan, 2024, pp. 11443-11450, doi: 10.1109/ICRA57147.2024.10610200.
  - “Concurrent work. There are two other concurrent works. [3] demonstrates agile behaviors by training task-specific policies and composing them using a high-level trained module but still relying on elevation maps.”
- Baines, R., Fish, F., Bongard, J. et al. Robots that evolve on demand. Nat Rev Mater 9, 822–835 (2024). https://doi.org/10.1038/s41578-024-00711-z
  - “For quadruped systems represented by rigid body dynamics simulations and placed within RL frameworks, zero-shot transfer of controllers to actual robots has been demonstrated in previously insurmountably complex and unstructured outdoor terrains162,163. We believe that an RL-based locomotion policy approach holds promise for AM (‘adaptive morphogenesis’) robots.”
- Y. Kim et al., "Not Only Rewards but Also Constraints: Applications on Legged Robot Locomotion," in IEEE Transactions on Robotics, vol. 40, pp. 2984-3003, 2024, doi: 10.1109/TRO.2024.3400935.
  - “Furthermore, RL controllers demonstrated high-speed dynamic locomotion of small-scale quadruped robots [8], [26], multiple gait transitions [27], [28], human-size bipedal robot locomotion [29], and even more agile movements, such as parkour [30] or environment interactions [31].”

Pros and Cons

PROS

- Learned approach is more robust to unknown environments than optimization-based control
- Modular approach can exploit strengths of different modules to chose optimal behavior (outperforming distilled policies where multiple steps are grouped in one model)
- Running models on robot is not computationally heavy
- Training can be done in simulation and transfers to robot

CONS

- Retraining one module often requires retraining the other ones linked to it
- Training time and tuning is time consuming due to the large number of models and their interdependencies
- Approach not yet tested for more complex terrain. Would probably need more low-level skills.

EPFL

Exam questions

Why is it necessary to retrain the navigation module if the locomotion module changes in an approach where these are separate?

- Retraining the navigation module is necessary as the capabilities of the locomotion modules might have changed and the optimal trajectory choice might therefore be different than before.

Why is it beneficial to have two models with different sizes and densities in the perception module?

- The use of two models is beneficial as it allows us to have one model with high resolution but small physical size and one with lower resolution but much larger spatial size. The use of a single high resolution would be less effective as it would be computationally expensive without any gain due to the lack of resolution of the sensors at longer distances.

Questions?