**Methodology**

**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

**Environment setup:**

Maybe? Just details of python/gymnasium code setup for the environment

**Game, action space, entities, states:**

Actions model can take (maybe reserved for section below)

Keys, doors, enemies, map size, what is success, what is failure

Termination states?

**dHRL, tables and network flow:**

Network hierarchy, root -> other subnetworks diagram

Tables of actions for each subtask, reward details

How does each subehaviour perform tasks? Navigation (pathfinding), Combat

**Training without PCG – baseline:**

For the baseline of the project, a model must be trained that has have no added techniques to enhance adaptability.

Talk about what this looks like, exactly what is our baseline model, and how will it be trained.

Exploration stochasticity?

This is opposed to the rest of the project, which will feature model’s trained using PCG.

**PCG:**

Non Perfect Maze generation using Kruskal’s algorithm.

* Kruskals generating a grid
* Union by rank

How are cycles added? What does a baseline map look like?  
How does the randomness of algorithm affect the layout.

Rough structure / difficulty (amount of edges).

Look up perfect generation too (min spanning tree)

**Training with PCG:**

Now we have PCG content, how are the models trained with it?

Varying degrees of PCG, what are they.

How many epochs?

How does the PCG enter the training loop, every iteration -> new map.

How are maps chosen, test set / train set

Exploration stochasticity?

**Added techniques:**

-Curricula learning, modifying the PCG algorithm

-CNNs

-Batch Normalisation / Regularisation

**Evaluation: (maybe this shouldn’t be in methodology?)**

Metrics, graphs