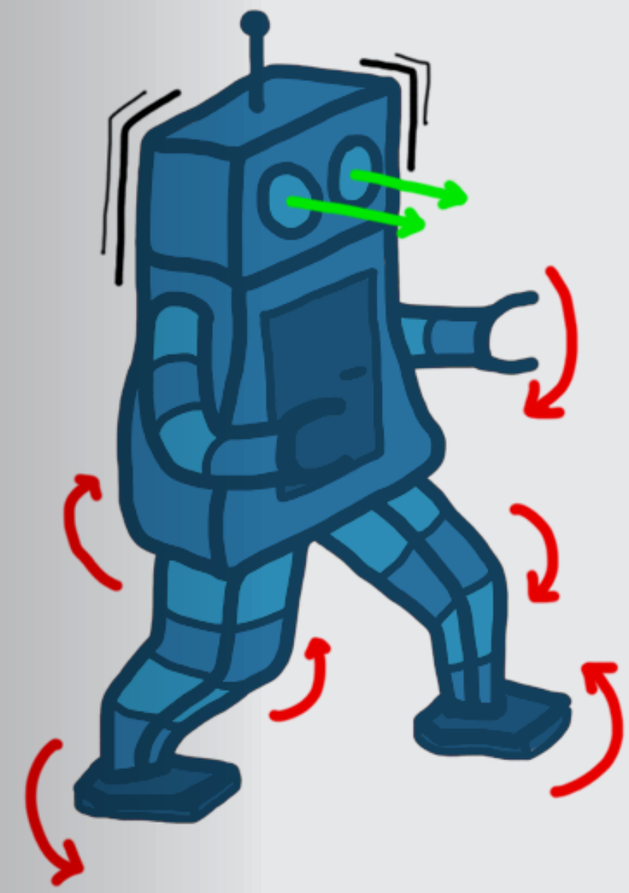
agent

environment



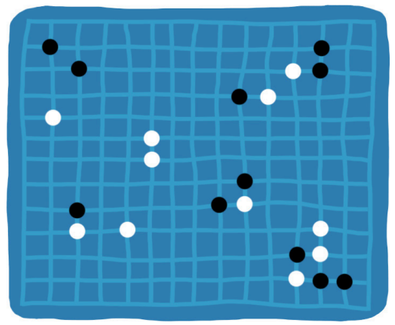
MATLAB による強化学習

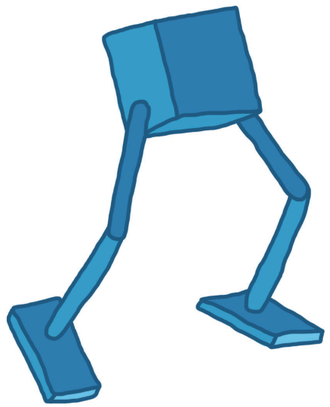
基礎の理解と環境の設定

MathWorks®

強化学習とは

> *強化学習とは、数値的な報酬信号の最大化を目的として、何をすべきか、状況に対してアクションをどのように対応付けるかを学習することです。学習器に実行するアクションは指示されません。その代りに、試行を通じて最大の報酬が得られるアクションを見つけ出す必要があります。*
>
> — Sutton および Barto、*強化学習: 概要*





強化学習 (RL, Reinforcement Learning) は、コンピュータープログラムをトレーニングすることで、ゲームで人間の世界チャンピオンに打ち勝つレベルに到達させることができました。

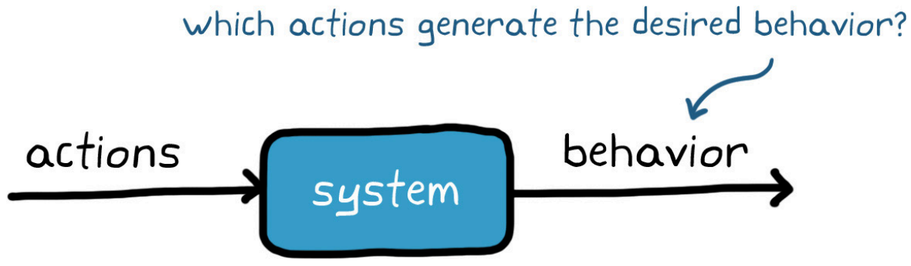
これらのプログラムは、大きな状態空間と行動空間、不完全な世界の情報が存在し、短期間のアクションが長期間の結果にどのような結果をもたらすかが不確実なゲームにおいて、最良のアクションを見つけ出します。

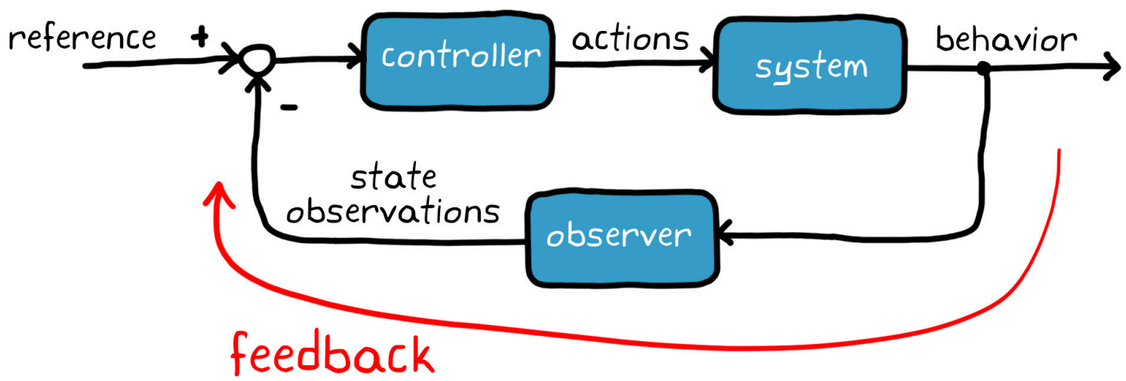
エンジニアは、実際のシステムのコントローラーを設計する際に、同様の課題に直面します。強化学習は、ロボットの歩行や、自律走行車の操縦などの複雑な制御問題の解決にも役立つでしょうか?

この eBook では、従来の制御問題と関連づけて強化学習を説明することでこの疑問に答え、強化学習問題の設定方法と解決方法を理解できるよう支援します。

制御の目標

大まかに言って、制御システムの目標は望ましいシステムの応答が得られるように入力 (アクション) を決定することです。





フィードバック制御システムでは、コントローラーは状態観測を使用してパフォーマンスを向上し、ランダムな外乱と誤差を修正します。エンジニアは、そのフィードバックをプラントと環境のモデルとともに使用して、システム要件を満たすコントローラーを設計します。

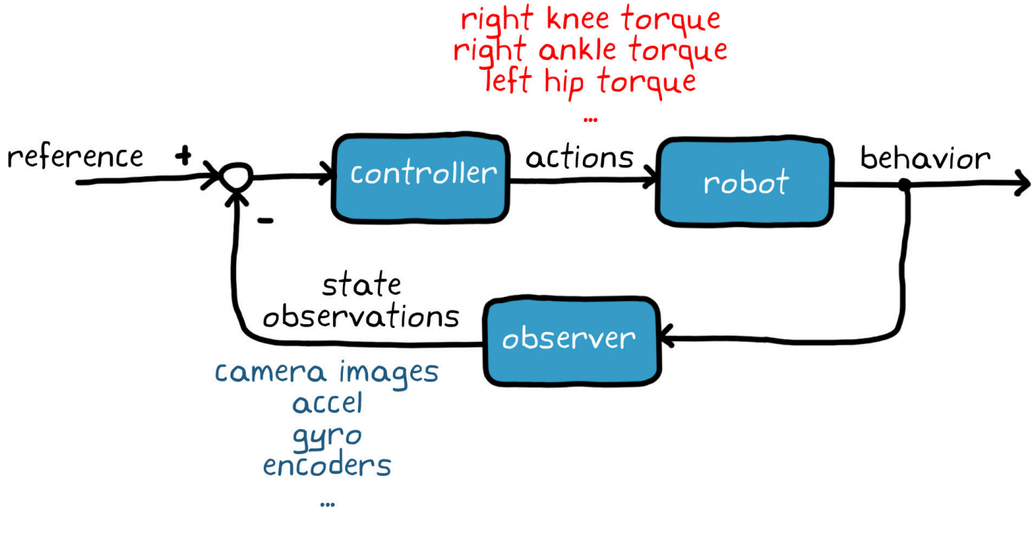
この概念は言葉にすると単純ですが、システムのモデル化が困難な場合、システムが著しく非線形である場合、または大きな状態空間と行動空間が存在する場合は実現が困難になります。

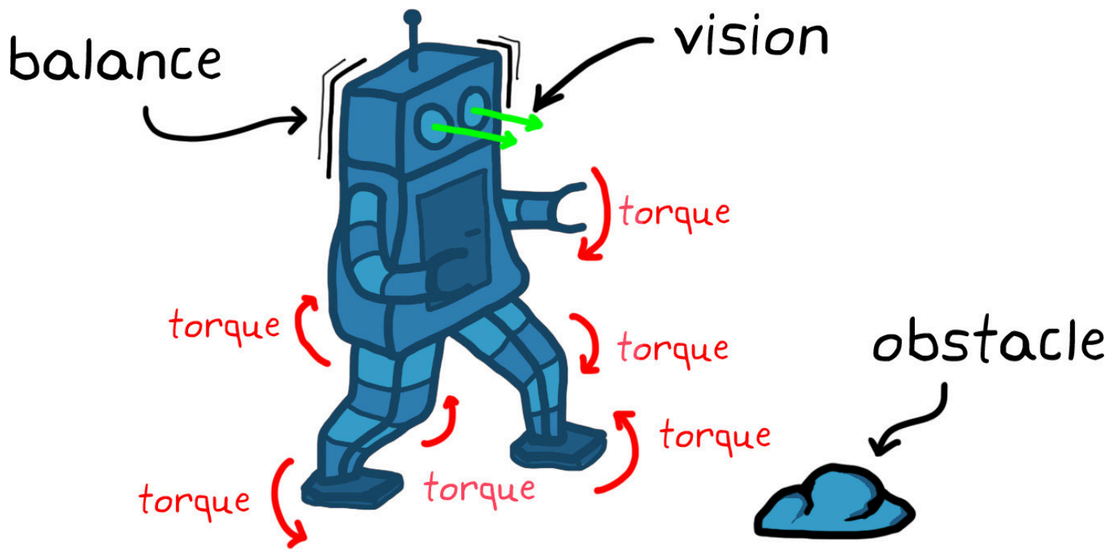
制御問題

複雑性がいかに制御設計問題を面倒にするかを理解するには、歩行ロボットの制御システムの開発を思い浮かべてください。

ロボット (つまりシステム) を制御するには、腕や脚の各関節を動かす多数のモーターへの命令が必要になります。

各命令が実行可能な 1 つのアクションになります。状態観測は、カメラのビジョンセンサ、加速度計、ジャイロ、各モーターの符号化器などの複数のソースから発生します。





コントローラーは、次のような複数の要件を満たさなければなりません。

- ロボットの歩行およびバランスの保持の達成に適したモータートルクの組み合わせの判断
- 避けるべき障害物がランダムに存在する環境での動作
- 突風のような外乱の排除

制御システム設計では、これらに対応するだけでなく、急勾配の斜面やところどころ凍った場所を渡る際のバランス維持などの要件への対応も必要になります。

制御ソリューション

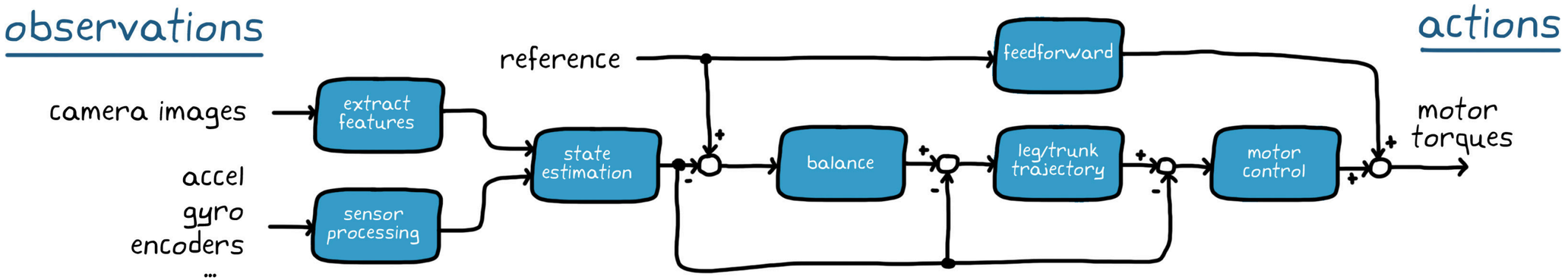
一般的に、この問題へのアプローチに最適な方法は、問題を小さな個別のセクションに分割し、それらを個々に解決することです。

たとえば、カメラのイメージから特徴を抽出するプロセスを構築できます。これはたとえば、障害の場所や種類であったり、グローバルな基準フレームにおけるロボットの位置であったりします。これらの状態を、他のセンサーから得られた処理済みの観測情報と組み合わせて、全体の状態評価を完成します。

推定された状態と基準がコントローラーに供給されます。多くの場合これはネストされた複数の制御ループで構成されます。アウターループは上位レベルのロボット動作 (バランスの維持など) を管理し、インナーループは下位レベルの動作と個々のアクチュエータを管理します。

これで全て解決でしょうか？いえ、それほど一筋縄では行きません。

ループは相互に作用するため、設計と調整を困難にします。また、これらのループの最適な構造を判断し、問題を分解することは容易ではありません。

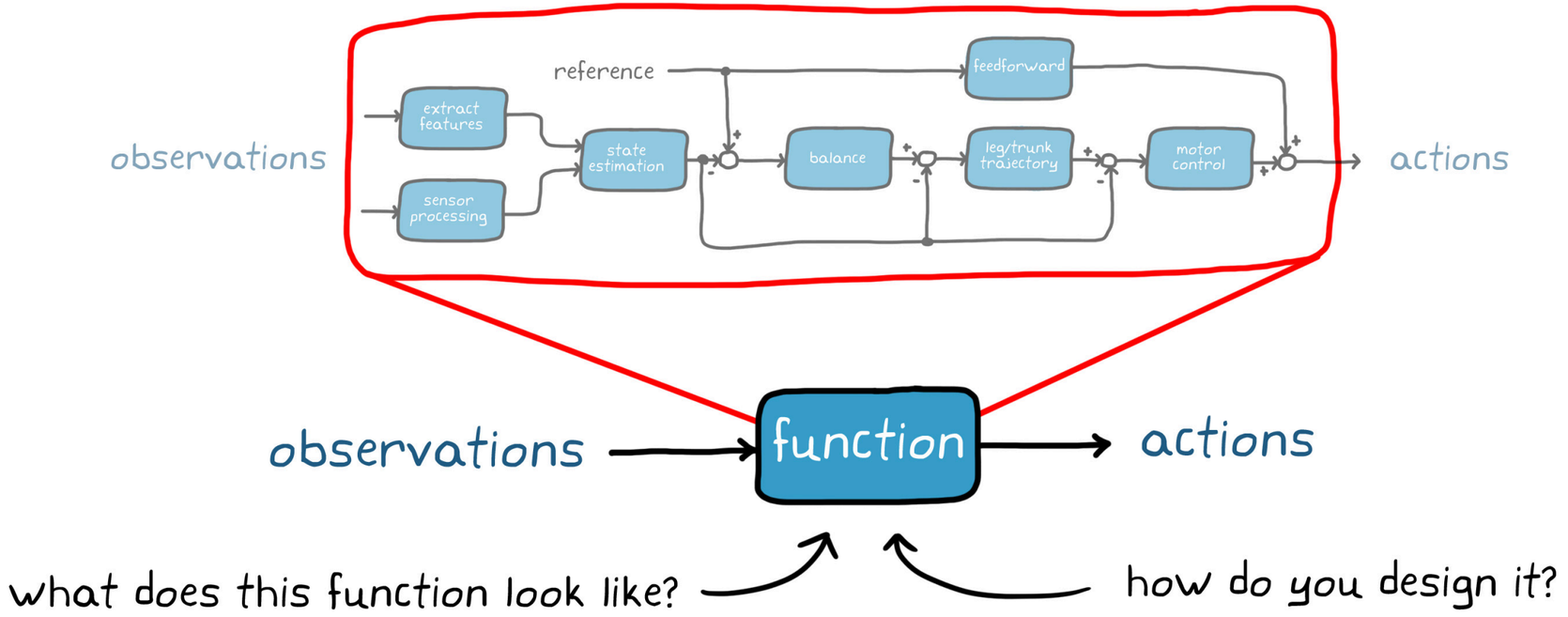


強化学習のメリット

これらのコンポーネントを個別に設計するのではなく、観測と下位レベルのアクションの出力のすべてを直接取り込む単一の関数に、あらゆるものを入力できるとしたらどうなるでしょうか。

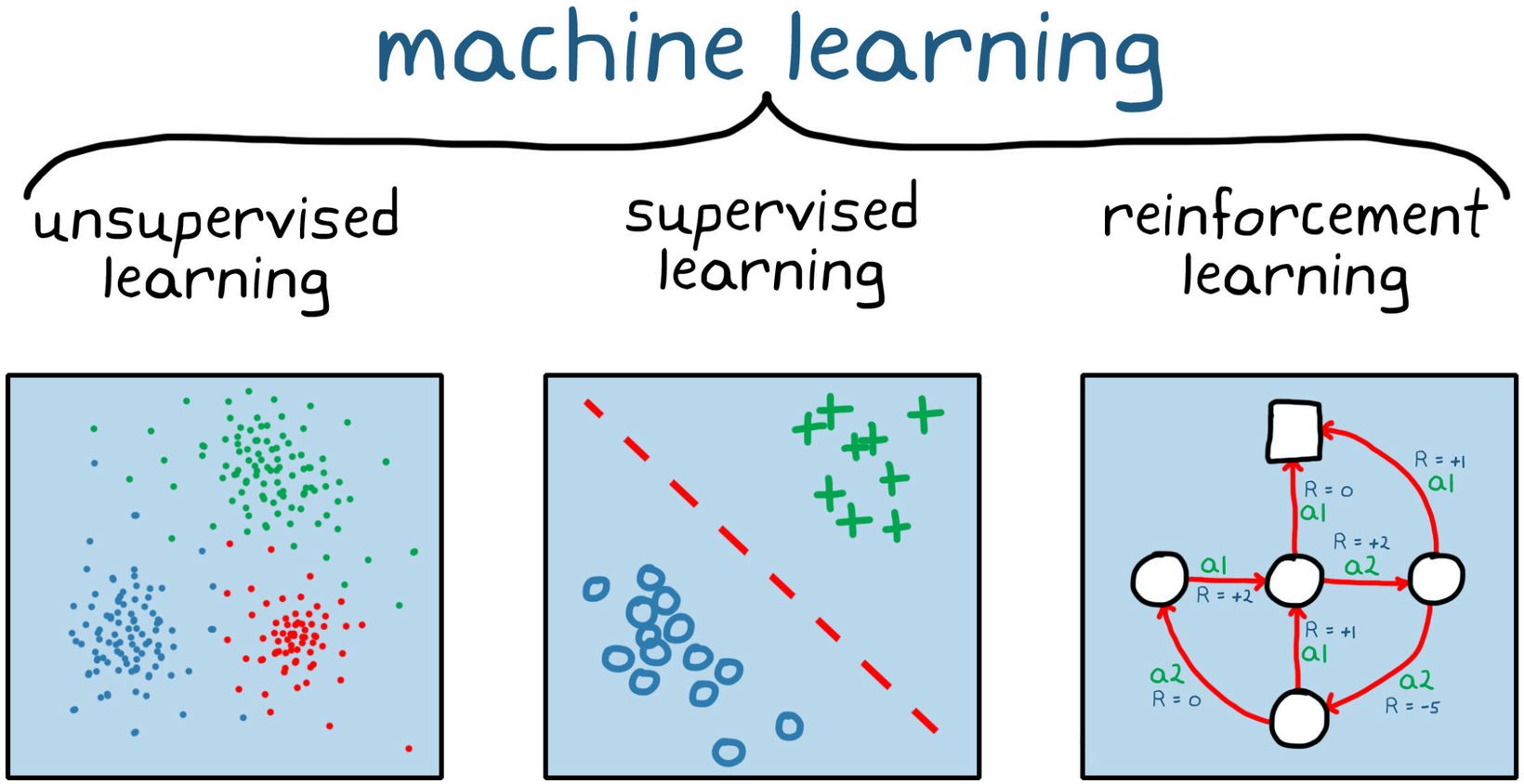
これでブロック線図は確実に簡易化されますが、この関数はどのようなものになり、どのように設計するのでしょうか？

この単一の大きな関数を作成することは、区分的なサブコンポーネントからなる制御システムの構築より困難に思えるかもしれませんが、これが強化学習が役立つところなのです。



強化学習: 機械学習のサブセット

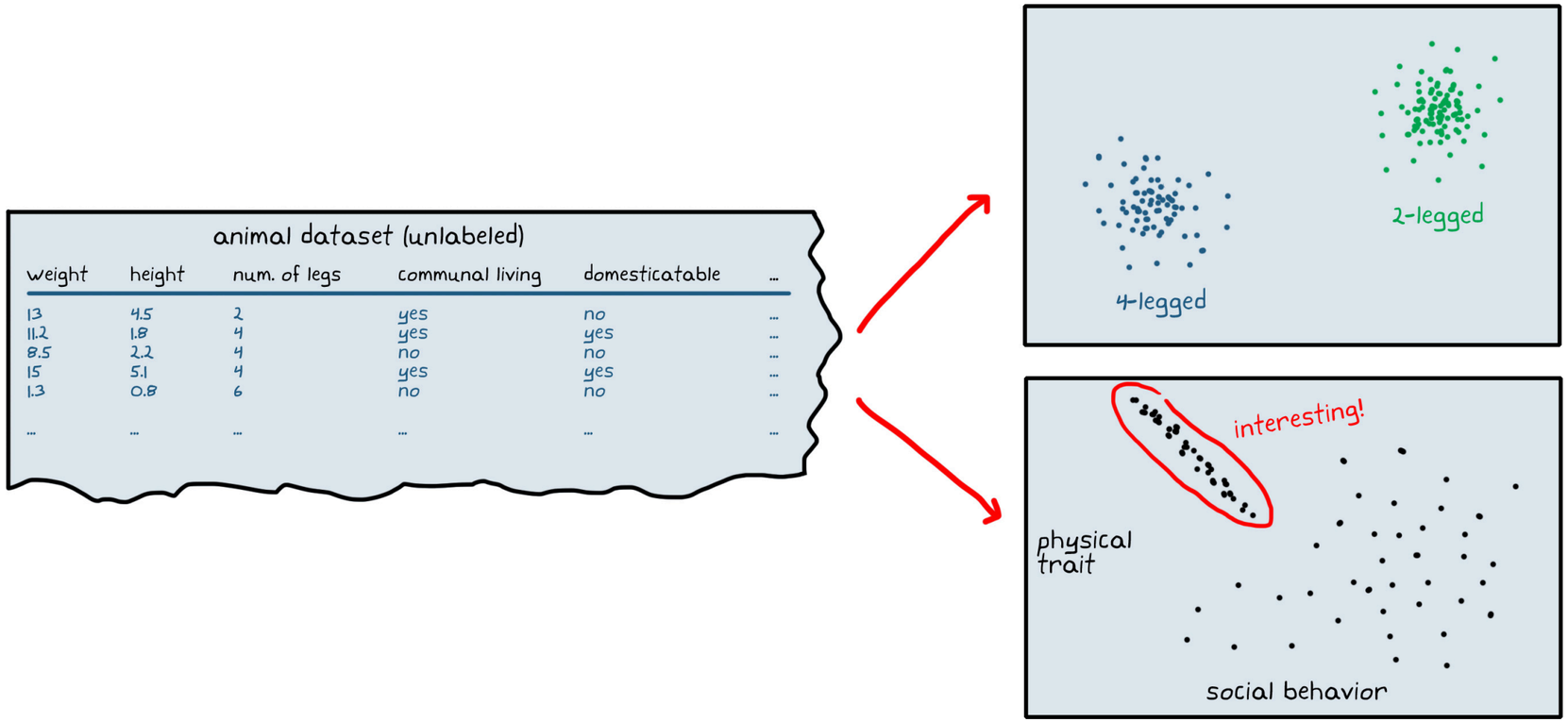
強化学習は、機械学習の主要 3 カテゴリのうちの 1 つです。この eBook では教師なし学習や教師あり学習の詳細は扱いませんが、これら 2 つと強化学習の違いを認識しておくことは重要です。



機械学習: 教師なし学習

教師なし学習は、分類やラベル付けが行われていないデータセットでパターンや隠れた構造の検出に使用されます。

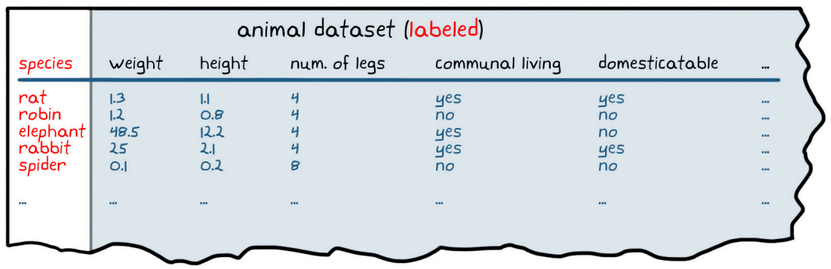
たとえば、10 万匹の動物の身体的特性と社会的傾向に関する情報があるとします。この場合、教師なし学習を使用して、動物をグループ化したり、類似した特徴でクラスタリングすることができます。これらのグループは、脚の数を基準にしたり、または事前に判明していなかった身体的形質や社会的動作の相関などのあまりはっきりしていないパターンを基準にすることができます。

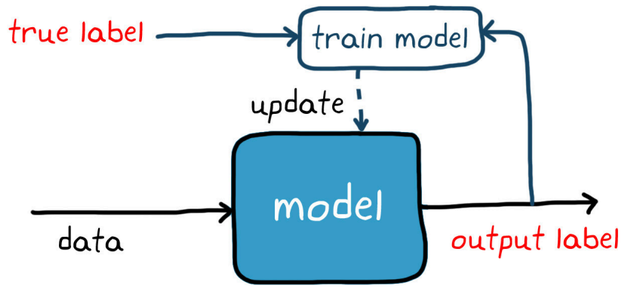


機械学習: 教師あり学習

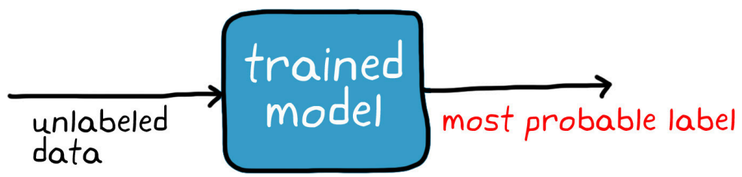
教師あり学習を使用し、特定の入力に対してラベル付けするようにコンピューターに学習させます。たとえば、動物の特徴のデータセット列の 1 つが種の場合、種をラベルとして扱い、残りのデータを数学的モデルへの入力として扱うことができます。

教師あり学習を使用してモデルに学習させ、データセット内の動物の特徴の各セットを正しくラベル付けすることができます。機械学習アルゴリズムは、モデルに種の推測とパラメータの微調整を体系的に実施させます。





信頼できるモデルを得るための十分なトレーニングデータがあれば、ラベル付けされていない新たな動物の特徴を入力すると、学習済みのモデルが最も有力なラベルをそれに付けます。

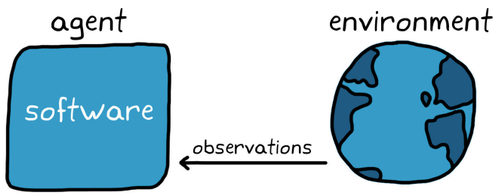


機械学習: 強化学習

強化学習はまったく異なるものです。他の 2 つの学習フレームワークは静的なデータセットを使用して機能しますが強化学習は動的な環境からのデータを処理します。また、強化学習の目標はデータのクラスタリングやラベル付けではなく、最適な解として最良のアクションの例を見つけることです。強化学習では、“エージェント” と呼ばれるソフトウェアを環境の探索、対話および学習に使用して、問題を解決します。

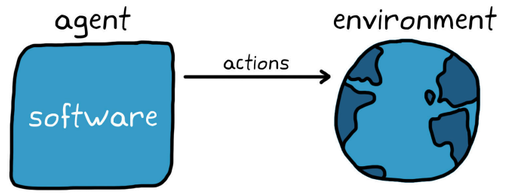
1

エージェントは、環境の現在の状態を観測できます。



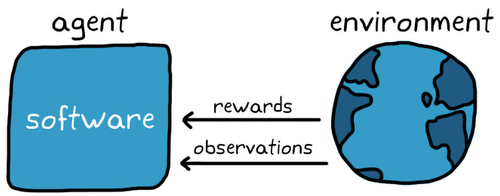
2

観測された状態から、エージェントはどのアクションをとるべきかを決めます。



3

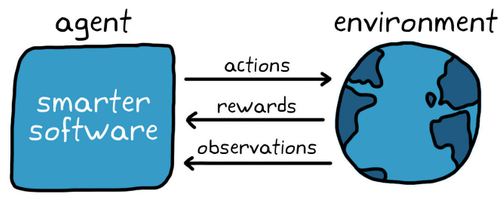
選択したアクションにより環境の状態が変化し、報酬が生成されます。エージェントはその両方を受け取ります。



4

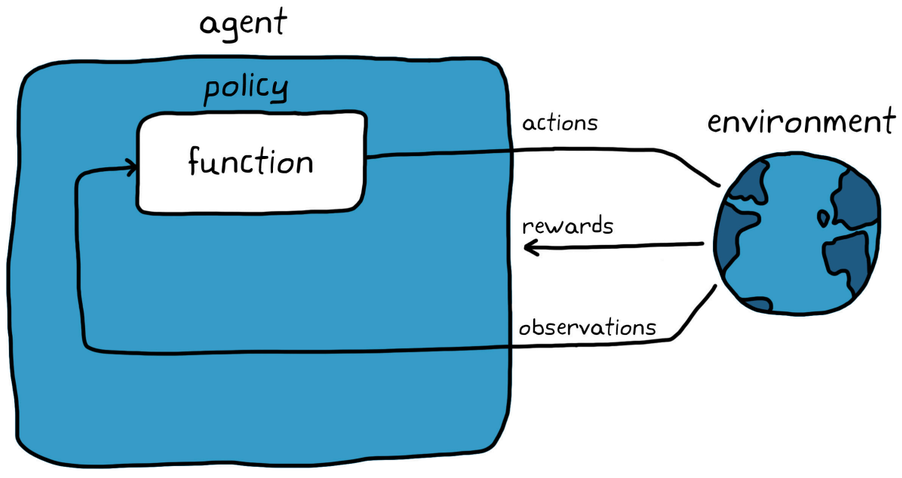
この新しい情報を使用して、エージェントは選択したアクションが良い手であり反復すべきか、または悪手であるため避けるべきかの判断が可能です。

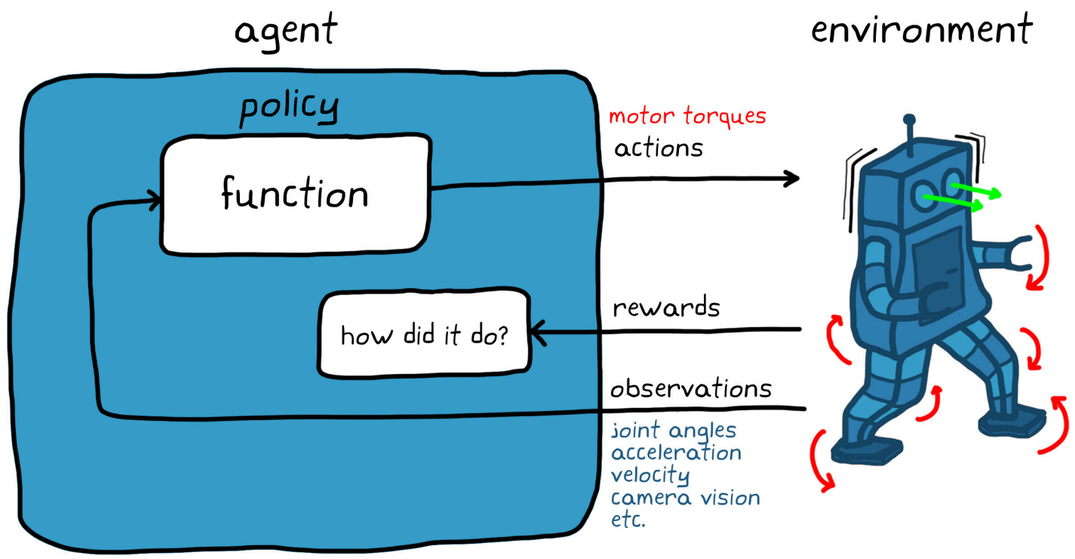
観測 - アクション - 報酬のサイクルを学習の完了まで続行します。



強化学習の構成

エージェント内に、状態の観測を取り込み (入力)、それをアクション (出力) に対応させる関数があります。これが前述の単一関数で、制御システムの個々のサブコンポーネントのすべての代わりとして置き換わります。
強化学習の用語で、この関数を “ポリシー” と呼びます。与えられた観測値から、ポリシーはとるべきアクションを決定します。





歩行するロボットの例では、観測情報は各関節の角度、ロボット胴体の加速度と角速度、そしてビジョンセンサーからの数万ピクセルからなる画像です。ポリシーはこれらすべての観測値を取り込み、ロボットの腕と脚を動かすモーターコマンドを出力します。

次に、環境によってアクチュエータ コマンドのある特定の組み合わせの好適度をエージェントに知らせる報酬が生成されます。ロボットが直立したまま歩行を続けることができれば、ロボットが地面に倒れた場合よりも報酬が高くなるようにします。

最適なポリシーの学習

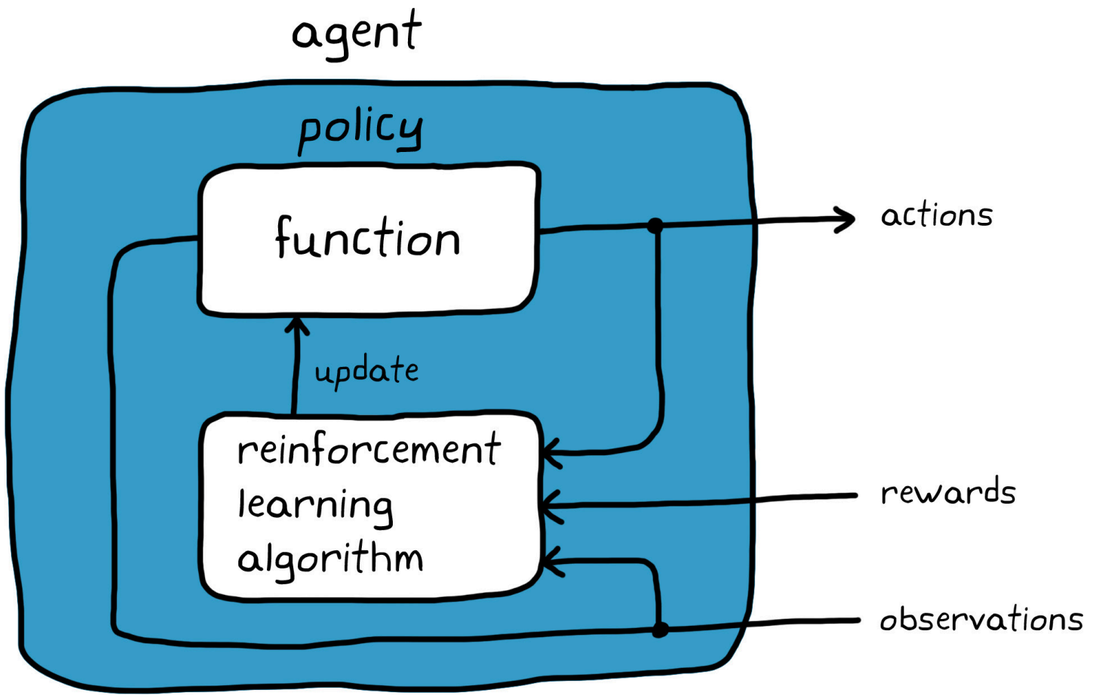
観測された各状態に応じて、適切なアクチュエータに正確に指令を出す完璧なポリシーを設計できれば、あなたの仕事は終わりです。

しかしながら言うまでもなく、多くの状況下では困難なことです。完璧なポリシーを見つけたとしても、環境は時間の経過につれて変化するため、静的なマッピングは最適なものではなくなります。

ここで登場するのが、強化学習アルゴリズムです。

強化学習アルゴリズムは、とったアクション環境からの観測、および収集された報酬に基づいてポリシーを変更します。

エージェントの目標は、強化学習アルゴリズムを使用して最善のポリシーを学習することであり、どのような状態でも環境と対話しながら常に最適なアクションをとるように、つまり長期にわたって最大の報酬を生成できるようにすることです。



学習の意味

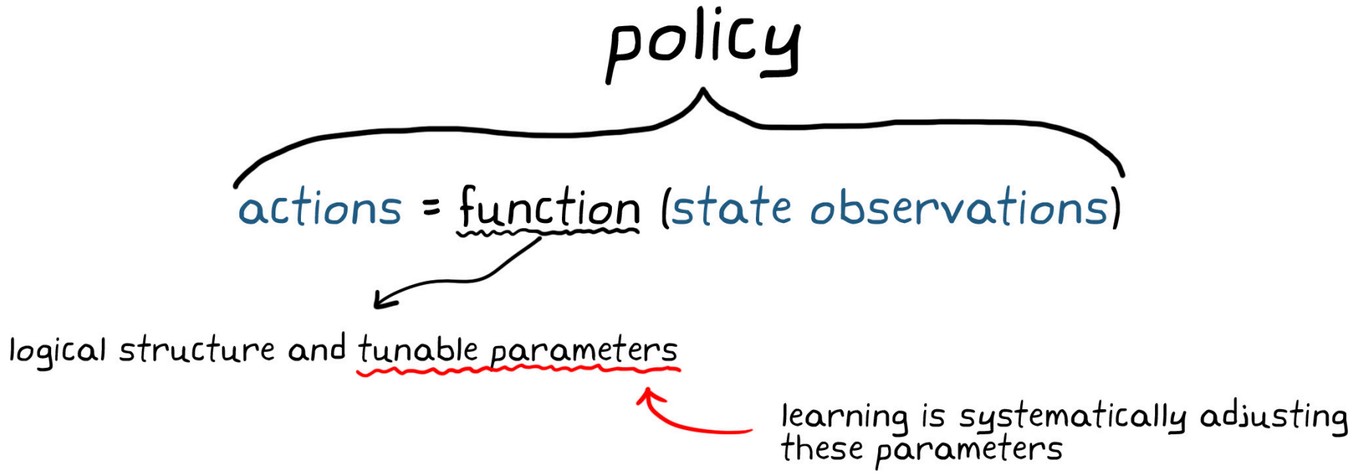
機械が学習するという意味を理解するため、ポリシーとは実際には何であるかを考えましょう。ポリシーとはロジックと調整可能なパラメーターで構成される関数です。

求められるポリシー構造 (ロジック構造) には、最適なポリシーを生成するパラメーターの集合が存在します。最適なポリシーとは、長期にわたって最大の報酬を生み出す状態からアクションへの写像です。

“学習” とは、パラメーターを体系的に調整して最適なポリシーに収束させるためのプロセスを指します。

このようにすることで妥当なポリシー構造の設定に専念すれば、関数を手動で調整することなく、適正なパラメーターを得ることができます。

のちほど取り上げるプロセスを通じて、関数のパラメータをコンピュータに学習させますが、ここではある種の試行錯誤とお考えください。



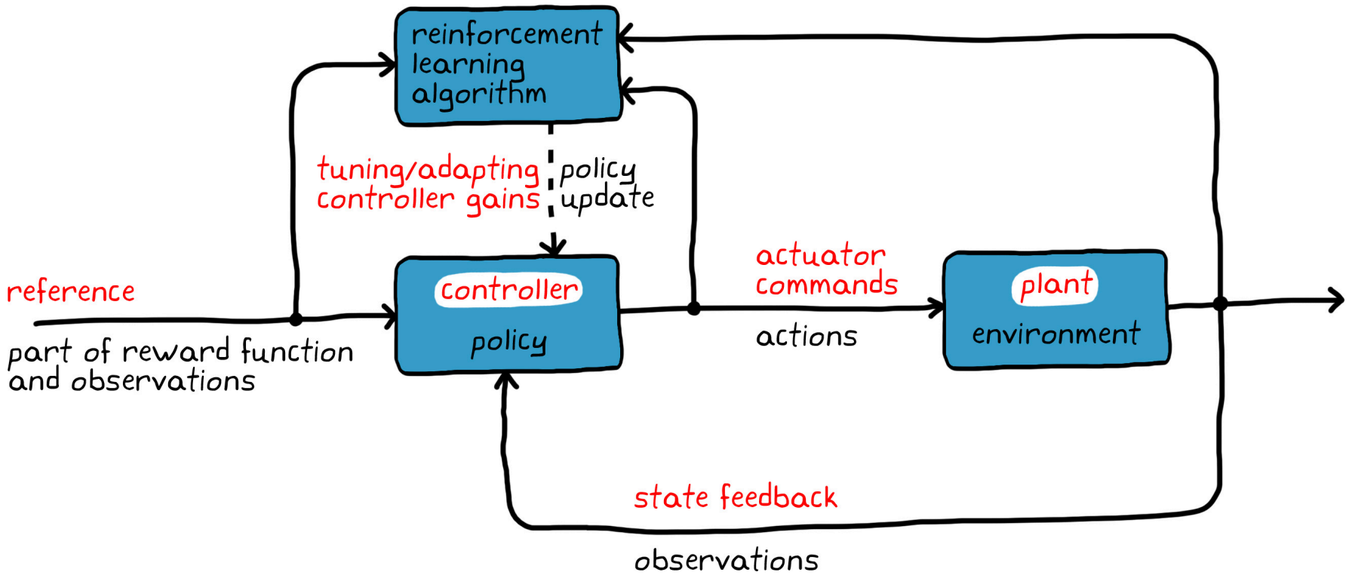
強化学習と従来の制御の類似点

強化学習の目標は、制御の問題と類似していますが、アプローチが異なり、同じ概念を表すにも関わらず使用する用語が異なります。

いずれの方法でも、必要なシステム動作を生成できるようにシステムへの正しい入力を判断する必要があります。

観測した環境状態 (プラント) を最良のアクション (アクチュエータ コマンド) に対応させるポリシー (コントローラー) を設計する方法を見出そうとしているのです。

状態フィードバック信号は、環境からの観測であり、基準信号は報酬関数と環境観測の両方に組み込まれます。



強化学習ワークフローの概要

一般的に、強化学習では 5 つの異なる領域を取り扱う必要があります。この eBook では、最初の領域である、環境の設定を中心に説明します。
このシリーズの他の eBook では、報酬、ポリシー、学習、配布について詳しく説明します。

1 エージェントが学習できる環境が必要です。環境内に何を含むべきか取捨選択し、さらに環境がシミュレータか、物理的なハードウェアを用意するかを選択する必要があります。

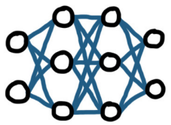
environment

2 エージェントに最終的に何をさせたいかを検討したり、エージェントにそれを行わせるための動機を与える報酬関数の設計をする必要があります。

reward

3 ポリシーを表現する方法を選択する必要があります。エージェントの意思決定部分を構成するパラメーターとロジックの構造化方法を検討します。

policy

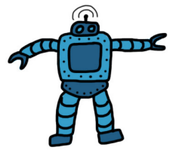


4 最適なポリシーパラメータを見つけるためのエージェントの学習アルゴリズムを選択する必要があります。

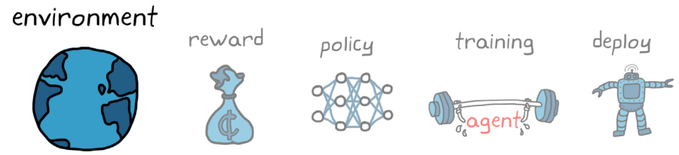
training

5 最終的に、ポリシーを現場に展開したり、その結果を検証するといったポリシーの活用が必要です。

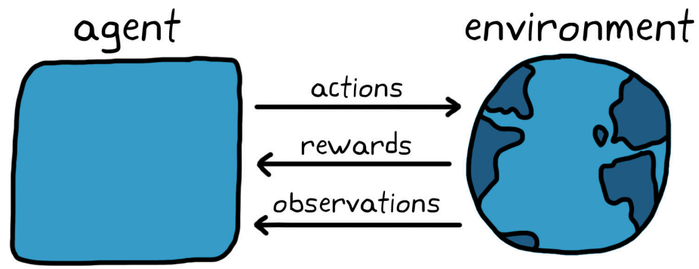
deploy

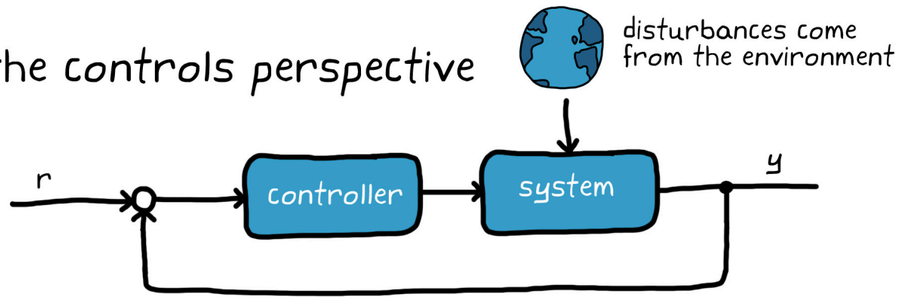


環境



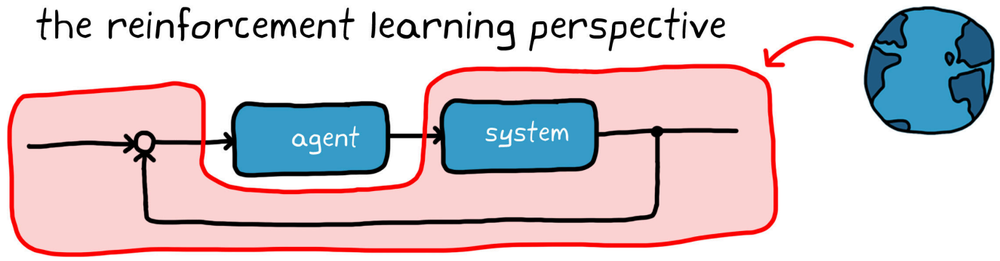
環境とは、エージェントの外側に存在するあらゆるものです。エージェントがアクションを送信する場所であり、報酬が生まれ、観測が行われる場所です。



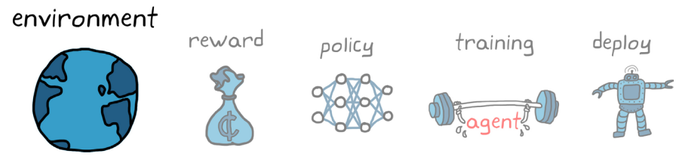


制御の視点で捉えるユーザーは、環境を、制御対象のシステムに影響を与える外乱であると考える傾向があるため、この定義は混乱を招くことがあります。

しかし、強化学習の用語では、環境はエージェント以外のすべてを意味します。これにはシステムのダイナミクスも含まれます。このように、実質的にシステムの大部分は環境に含まれます。エージェントは、アクションを生成し、学習を通じてポリシーを更新する小さなソフトウェアにすぎません。



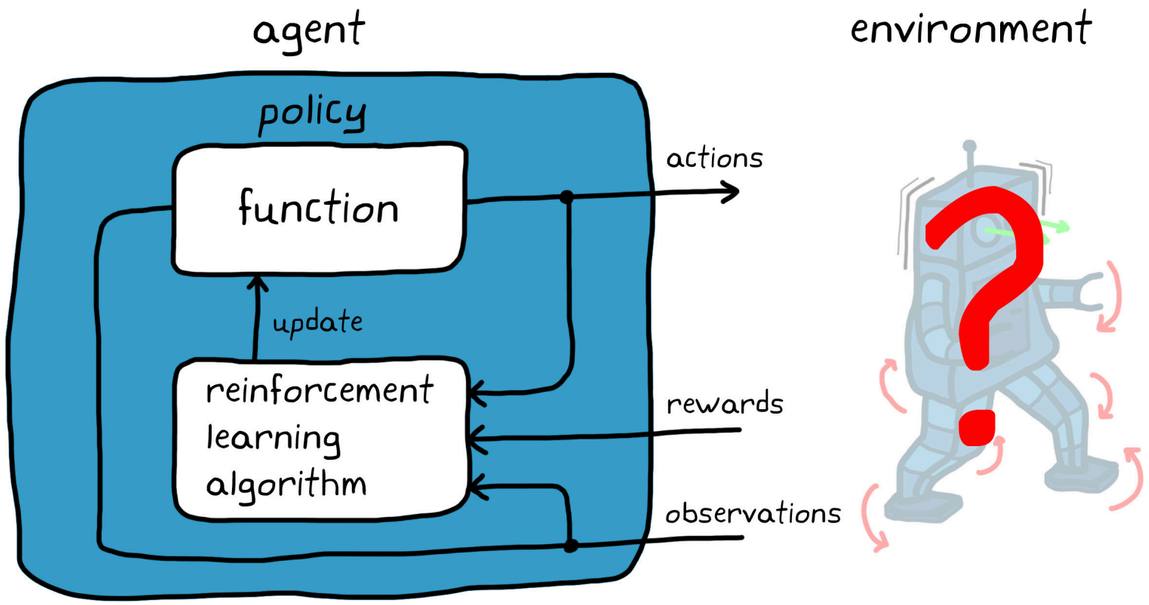
モデルに依存しない強化学習



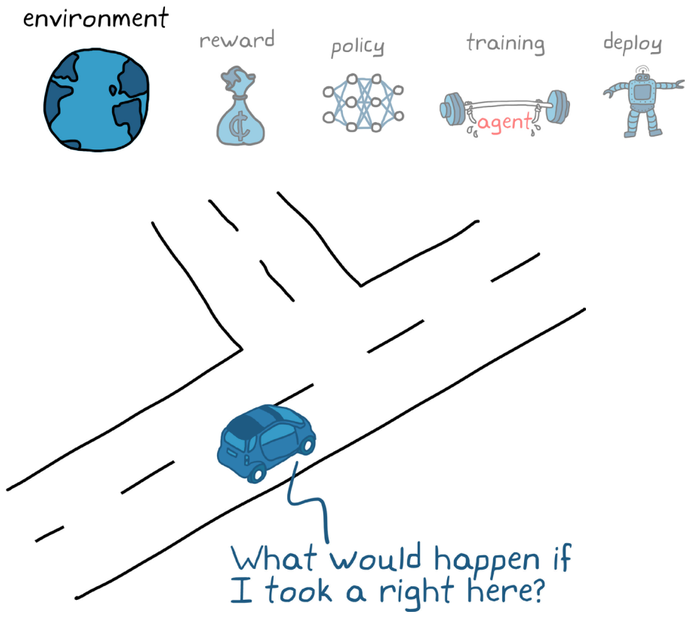
強化学習が非常に強力である理由の 1 つは、エージェントが環境について何も認識している必要がないことです。しかし、環境との対話方法の学習はできます。たとえば、エージェントは歩行ロボットのダイナミクスや運動学を認識する必要はありません。関節がどのように動くかや脚の長さを知らなくても、最大の報酬を得る方法を見つけ出すことができます。

これは、“モデルに依存しない強化学習” と呼ばれます。

モデルに依存しない強化学習を使用し、強化学習エージェントをシステムに配置すれば、エージェントは最適なポリシーを学習できます (ポリシーに観測、報酬、アクション、および十分な内部状態へのアクセスを与えていると仮定します)。

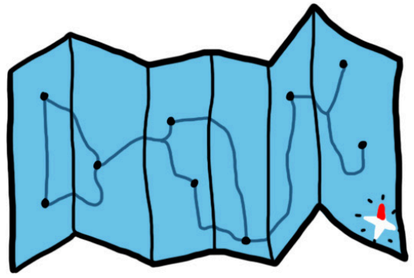


モデルベースの強化学習



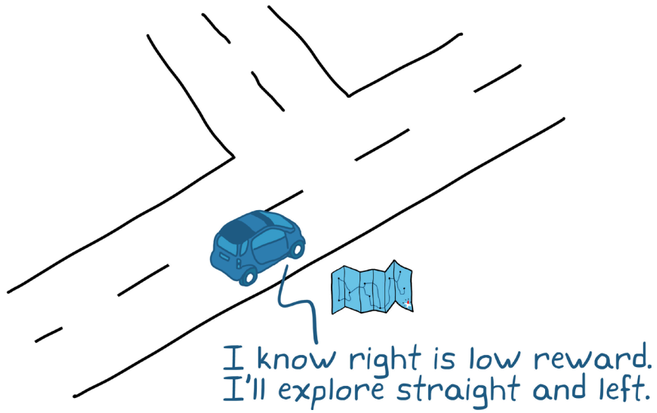
モデルに依存しない強化学習にも問題があります。エージェントが環境について何も知らなければ、エージェントが最大の報酬を受ける方法を見つけるために、状態空間のすべての領域を探索しなければなりません。

これはつまり、エージェントは学習プロセス中に、低報酬の領域の探索に時間をかけなければならないことを意味します。

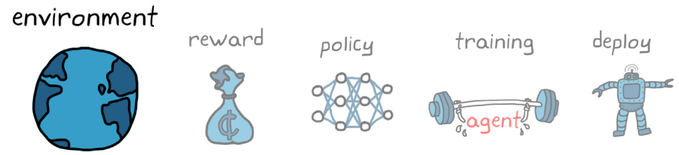


しかし、状態空間の一定の部分は、探索する価値がないことがわかっているとします。環境のモデル、または環境の部分を提供して、エージェントにこの知識を提供します。

モデルの使用により、エージェントは、実際にアクションを起こさずに、環境の一定の部分を調べることができます。不要な部分を避け、他の部分を探索するようにすることで、モデルで学習プロセスを補完できます。



モデル非依存 vs モデルベース



“モデルベースの強化学習” は、モデルを使用してエージェントをガイドし、低報酬であると認識されている状態空間の領域を避けることができます。そのため、最適なポリシーを学習する時間を短縮できます。

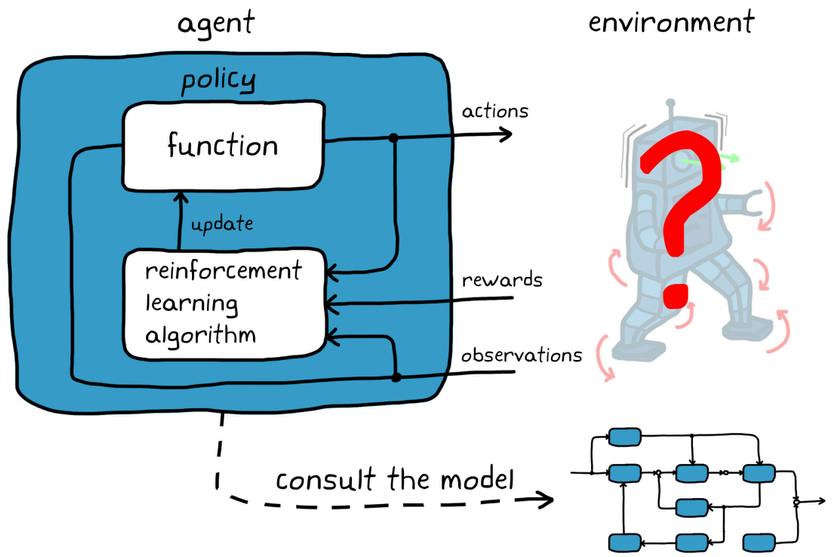
そもそも、エージェントが低報酬の状態に到達しないため、そのような状態での最良アクションを学習する必要がありません。

モデルベースの強化学習では、環境モデル全体を知る必要はありません。既知の環境部分のみをエージェントに提供できます。

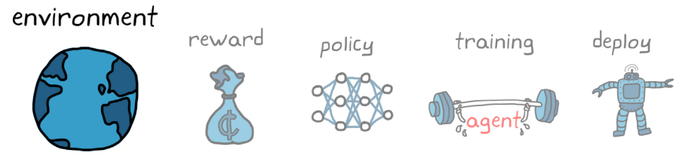
“モデルに依存しない強化学習” は、より一般的なケースであり、この eBook の後続部分で説明します。

モデルなしの強化学習の基礎を理解すると、モデルベースの強化学習の理解も進みます。

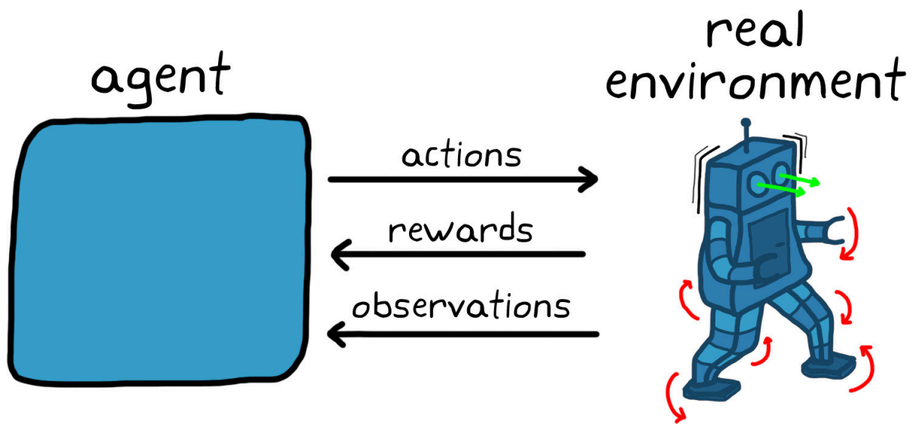
モデルに依存しない強化学習は現在広く使用されていますが、これは単純なものであってもモデルの開発が困難な場合に問題の解決に使用したいと考えられているためです。一例として、ピクセル観測からの車やロボットの制御が挙げられます。ピクセルの明暗度と車やロボットのアクションの関連は、多くの場合、直観的にわかるものではありません。



実環境 vs シミュレーション環境



エージェントは環境との対話から学習するため、エージェントには実際に環境と対話する方法が必要です。これは実際の物理的環境の場合もシミュレーションの場合もあり、この 2 つのいずれを選択するかは状況によって異なります。

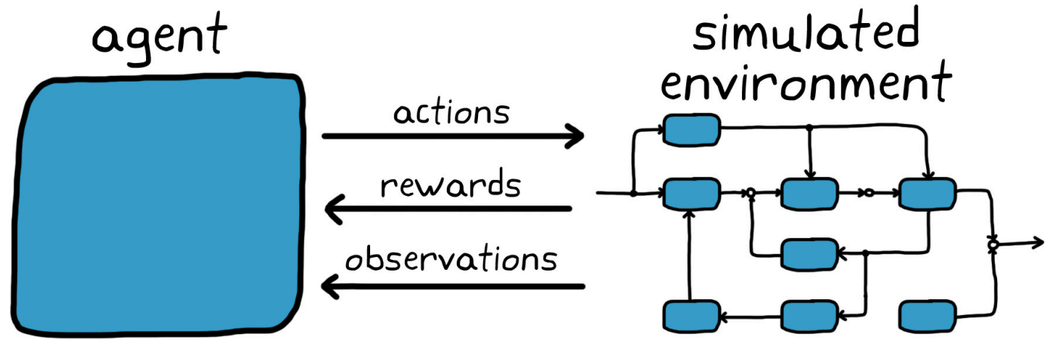


実環境

精度: 実環境ほど、環境を完全に表現できるものはありません。

簡潔性: モデルを作成したり、検証したりするために時間を費やす必要がありません。

必要性: 環境が継続的に変化する場合や、正確なモデル化が困難な場合は、実際の環境で学習させることが必要な場合もあります。



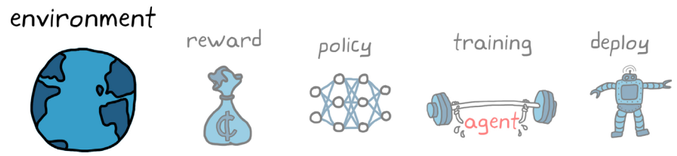
シミュレーション

スピード: シミュレーションは実際よりも高速に実行でき、また並列化も可能であり、低速な学習プロセスを高速化できます。

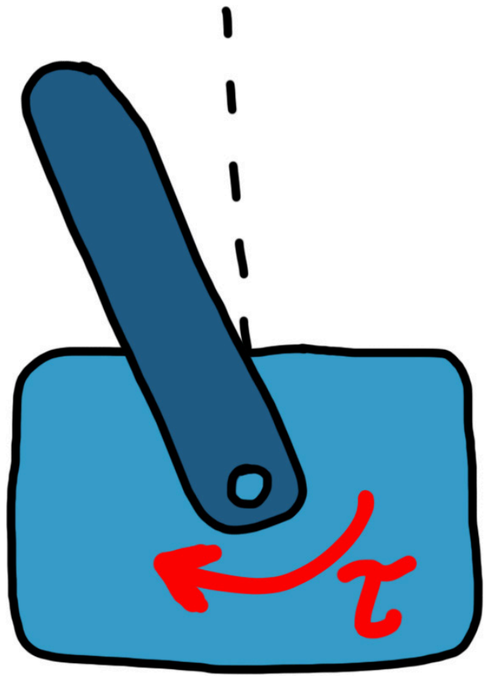
条件のシミュレーション: テストが困難な状況では、モデル化の方が容易です。

安全性: ハードウェア破損のリスクはありません。

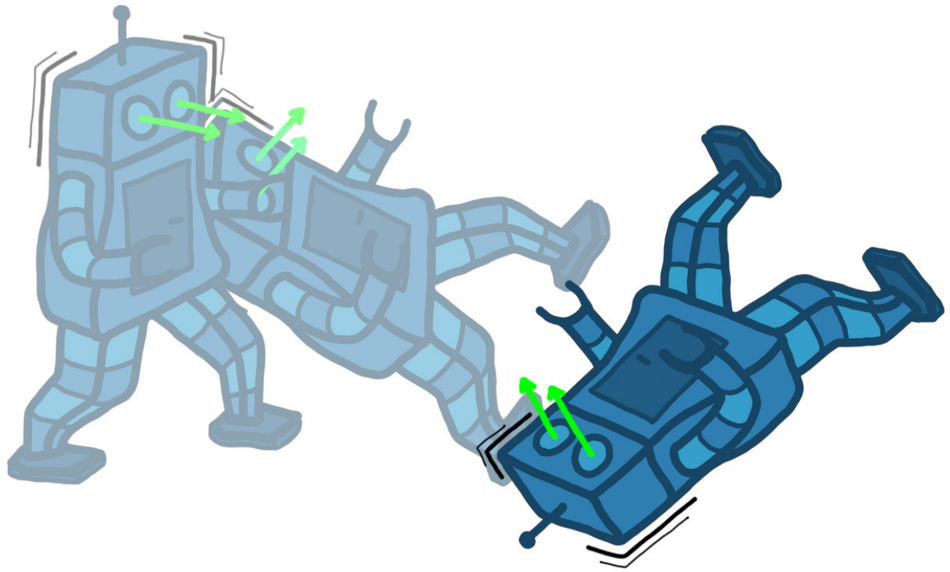
実環境 vs シミュレーション環境



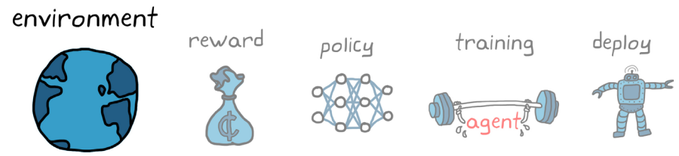
たとえば、実際に振子を設置して動かすことで、エージェントに倒立振子のバランスを保つ方法を学習させることができます。ハードウェアがそれ自体や他のものを破損する可能性は低いため、これは良い解決策と言えるでしょう。状態空間と行動空間が比較的小さいため、トレーニングにもそれほど時間がかからないでしょう。



しかし、歩行ロボットのケースでは、これは適したアイデアではないかもしれません。トレーニングの開始時点で、ポリシーが十分に最適化されていなければ、歩行するどころか、脚を動かす方法を学習するまでに、ロボットは何度も転倒したり壊れたりするでしょう。これはハードウェアを破損させるだけでなく、毎回ロボットを起こすために非常に時間がかかります。よって理想的とは言えません。



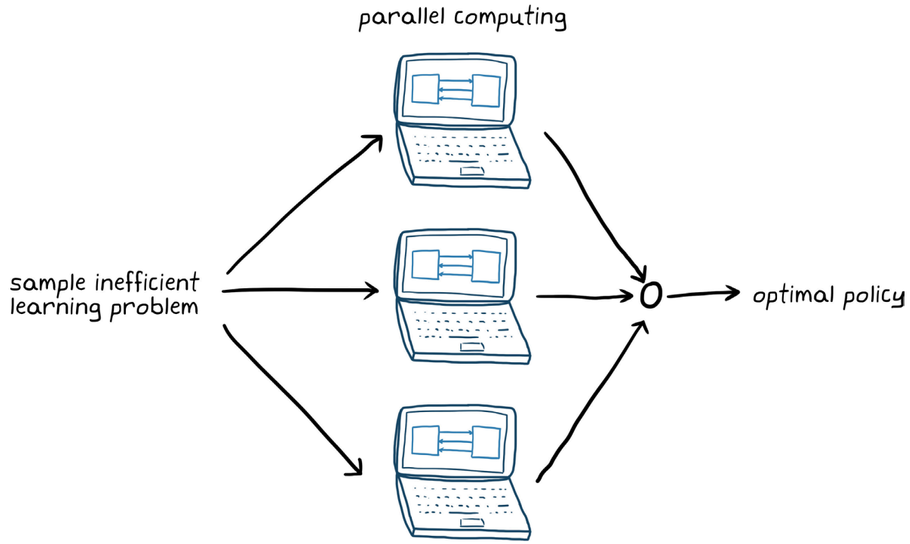
シミュレーション環境のメリット

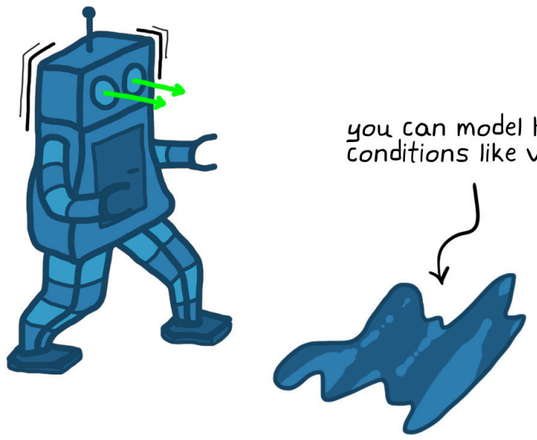


シミュレーション環境は、エージェントのトレーニングに最も一般的な方法です。制御問題に関するメリットの 1 つは、一般的に従来の制御設計ではシステムと環境のモデルが必要とされるため、通常すでに好適なモデルがあることです。MATLAB® または Simulink® で構築されたモデルがすでに存在する場合は、既存のコントローラーを強化学習エージェントに置き換え、環境に報酬関数を追加して、学習プロセスを開始できます。

学習とは、試行、エラー、修正などの多くのサンプルを必要とするプロセスです。このため、1 つの最適な解決策に収束させるまでに何千あるいは何百万もの事例が必要となる場合があるため、非常に非効率的です。

環境のモデルは実際よりも高速で実行でき、また多くのシミュレーションを並列的に実行して速度を上げることができます。この両方のアプローチにより、学習プロセスの速度を上げることができます。





シミュレーションの状況を使用すると、エージェントを実世界で使用するよりも、制御がはるかに容易になります。

たとえば、ロボットはさまざまな異なる路面を歩行しなければならないことがあります。氷のような低摩擦の路面を歩くシミュレーションは、実際の氷を使ってテストするよりもずっと簡素になります。さらに、低摩擦の環境でエージェントをトレーニングすると、あらゆる路面でのロボットの直立の維持に実際に役立ちます。シミュレーションを使用することで、トレーニング環境の創出を強化できます。

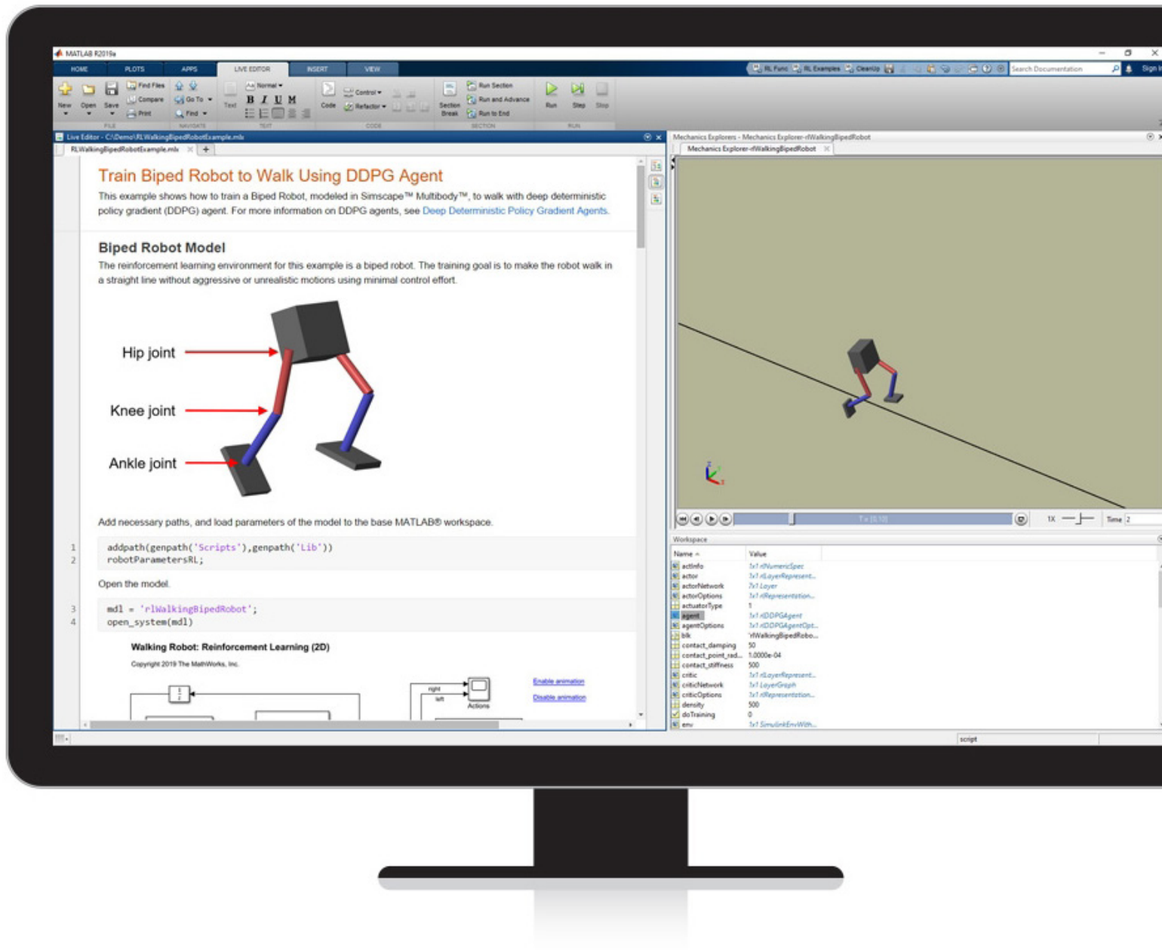
MATLAB および Simulink を使用した強化学習

Reinforcement Learning Toolbox には、強化学習アルゴリズムを使用したポリシーのトレーニングのための関数とブロックがあります。これらのポリシーを使用して、ロボットや自律システムなどの複雑なシステムのためのコントローラーと意思決定アルゴリズムを実装できます。このツールボックスを使用すると、MATLAB で表現された環境との対話を可能にして、または Simulink モデルを使用して、ポリシーに学習させることができます。

たとえば、MATLAB で強化学習環境を定義するには、提供されたテンプレートスクリプトとクラスを使用し、環境ダイナミクス、報酬、観測、アクションを用途の必要に応じて変更します。

Simulink では、制御と強化学習の問題の解決に必要となる頻度が高い、多様なタイプの環境をモデル化できます。たとえば、車両運動や飛行力学、Simscape™ を使用した各種の物理システム、System Identification Toolbox™ によって測定データから近似化されたダイナミクス、レーダー、ライダー、IMU のようなのセンサーなどをモデル化できます。

mathworks.com/products/reinforcement-learning



関連情報

見る

強化学習とは? (14 分 05 秒)

環境と報酬の理解 (13 分 27 秒)

歩行ロボットのモデリングおよびシミュレーション (21 分 19 秒)

調べる

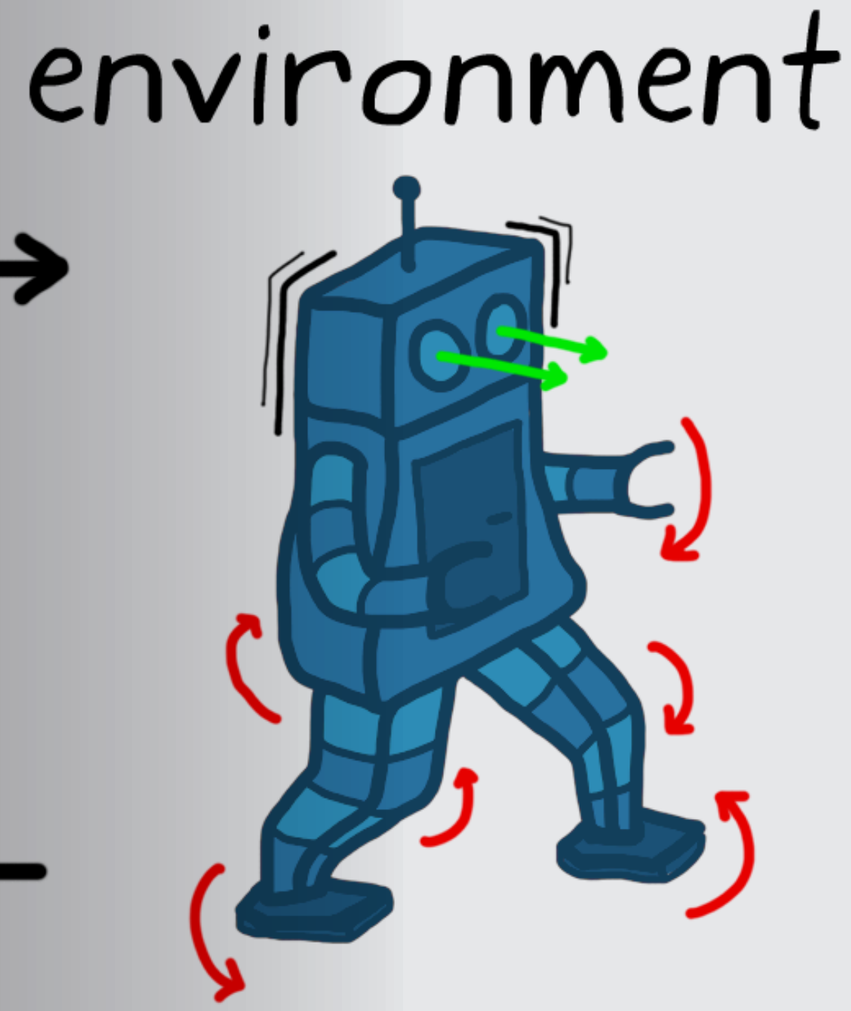
Reinforcement Learning Toolbox 入門

MATLAB での環境作成

Simulink での環境作成

Simulink での飛行力学のモデリング

Simulink での完全車両運動のモデリング



MathWorks