ステージ	NVIDIAワークフロー	非NVIDIAワークフロー
ステージ1: 仮想3D環境の構築	対象環境のデジタルツインを構築し、シミュレーションやデータ生成を行う段階です。NVIDIA Omniverseプラットフォームには、物理的なリアリズムを備えた3D環境構築のためのAPIやツールが用意されています。Universal Scene Description（OpenUSD）とRTXレイトレーシングにより、工場内、住宅、街区などを、正確な形状、光源、物理特性で表現できます。また、シミュレーションに必要な高負荷なグラフィックス計算はNVIDIA OVXサーバが処理します。さらに、NVIDIAのfVDB（スパースで大規模な高性能空間知能のための深層学習フレームワーク、現時点ではアーリーアクセス）により、巨大な3Dデータセットを対象に学習を行うことが可能です。これらのツールを活用し、エージェント（ロボットや車両）の行動開発と試験を行うための詳細な仮想空間を構築します。	この段階では、物理ベースのシミュレーターを使用し、デバイスが稼働する環境を再現します。ROS 2が、シミュレーション内でのセンサーやアクチュエーターの接続（ドライバーやメッセージ通信など）を担います。Gazeboは代表的なオープンソース3Dシミュレーターで、屋内外の環境と現実的なセンサー物理を再現します。自律走行車のシナリオではCARLAが広く利用されており、実都市環境に近い交通や道路状況、車載センサーを再現できます。ドローンや航空機の場合、MicrosoftのAirSimが一般的です。例えば、倉庫のCADモデルをGazeboに取り込み、ROS 2ドライバーを活用してLiDARやカメラを備えた仮想ロボットを操作するなど、環境とロボットのデジタルツインを構築します。実環境での試験に移る前の開発と評価を行うサンドボックスの役割を果たします。
ステージ2: 合成データ生成	仮想環境が整った段階で、学習用に大量のラベル付きセンサーデータを生成します。実世界データが不足している、または制約がある場合の補完となります。NVIDIA Omniverse Replicator SDKは、照明、物体配置、テクスチャなど仮想シーンの要素をランダム化することで、合成画像や点群データを生成します。多様なバリエーションをレンダリングすることで、幅広いシミュレーションセンサー出力が得られます。NVIDIA Cosmosプラットフォームは、フォトリアリスティックな合成データ生成パイプラインを提供し、レンダリングシーンから実写に近い画像を生成できます。自動的に生成される多数のラベル付きデータ（カメラ画像の正解アノテーションやLiDARスキャンの物体識別など）が、認識モデルや認知モデルの学習に活用されます。	視覚や認識モデルを学習するため、シミュレートされた環境でラベル付きの合成データを生成します。GazeboやCARLAでは、照明、テクスチャ、天候などを体系的に変えるドメインランダム化を用いて、堅牢な学習データを作ります。Unity（Machine Learning Agents: ML-Agents対応）やUnreal Engineなどのゲームエンジンも合成シナリオ生成に利用でき、強化学習（RL）にも対応します。基本的な考え方はNVIDIA Replicatorと同様で、多様な学習データを得ることにあります。例えば、Gazebo内の物体配置をランダム化し、正解ラベル付きカメラ画像を大量に取得し物体検知モデルの訓練に利用します。Unity ML-Agentsでは、学習エージェントを仮想環境内に配置し、RLで経験を生成できます。
ステージ3: モデル学習と検証	合成データと実世界データを用いて、デバイス上で動作するAIモデルを学習します。NVIDIA DGXプラットフォームによって深層学習の高速化が可能です。TensorFlowやPyTorchに加え、事前学習モデルや効率的な学習パイプラインを備えたTAO Toolkitも利用できます。初期学習後はシミュレーションで検証を行います。NVIDIA Isaac Sim（Omniverse上のロボティクスシミュレーター）で仮想シナリオを実行し、期待通りに動作するか確認します。NVIDIA Isaac GymやIsaac Cortexを用いると、仮想環境における強化学習で制御ポリシーを鍛えることができます。「学習→Isaac Simでの検証」の反復により、実機投入前に問題を解消していきます。	収集した合成データと実世界データを用いてAIモデルを学習します。PyTorchやTensorFlowが一般的で、ワークステーションやクラウドGPUで高速化します。OpenAI Gym、Stable Baselines3、RLlibなどの強化学習ライブラリはROS 2やシミュレーターと統合でき、試行錯誤による制御ポリシーの学習が可能です。例えば、Gazebo内のROS 2ロボットアームをOpenAI Gym環境と接続し、把持動作のRL学習を行います。MuJoCoのような物理エンジンは、正確なダイナミクスを伴う制御モデルの学習に役立ちます。高次の推論や自然言語処理（NLP）が必要な場合、Hugging Face Transformersなどの事前学習モデルを組み込むことも可能です。学習の進行に合わせ、GazeboやCARLAでタスク遂行能力（衝突回避や物体認識など）を確認し、モデルとソフトウェア全体を改善します。
ステージ4: エッジハードウェアへのデプロイ	シミュレーションで良好な性能が得られた段階で、最適化されたモデルを実機にデプロイします。NVIDIAはJetsonモジュール（ロボット、ドローン、IoTデバイス向け）やDRIVE AGX（自動運転向け）など、エッジ推論専用のハードウェアを提供しています。TensorRTやONNXへの変換を通じて推論が高速化され、モデルはロボットのソフトウェアスタックに統合されます。ROSとの連携にも対応しており、実機試験が行われます。最終的なシステムは、Jetson AGX Xavier上でCUDAとTensorRTにより認識モデルを稼働させ、Isaac ROSなどでパイプライン全体を制御する構成となります。	開発が完了したシステムを実デバイスで動作させます。ロボットには通常、Raspberry Pi、NVIDIA Jetson、Intel NUCなどの組み込みコンピュータが搭載され、ROS 2がランタイムとなります。モデルはOpenVINOやTensorRTで最適化し、ONNXで汎用性を確保してデバイスに搭載します。実センサーとアクチュエーターを用いた試験で、期待通りに動作するか確認します。ファームウェアの保護、クリティカルな処理用リアルタイム制御、フェイルセーフの実装など、運用を見据えた対策が必要です。クラウドとの連携により、更新や処理のオフロードにも対応可能で、この段階で自律的なエッジAIエージェントとして稼働します。オープンソース中心の構成は、必要に応じてコンポーネントを柔軟に切り替えられる利点がありますが、複数要素の統合が必要になります。