第二章 Python 統合開発環境

作成：Weiwei Wan
修正：Weiwei Wan, Takuya Kiyokawa

2024 年度 春～夏学期

Python を用いた WRS の開発では，統合開発環境（IDE）の使用を推奨します．IDE を使うと，コマンドライン上で直接プログラムを書かずに済み，コード補完や Git 連携などの機能を利用できるため，効率的に開発できます．Python の IDE には，PyCharm，Visual Studio Code，Spyder，Jupyter Notebook などがありますが，その中でも PyCharm は Python の開発コミュニティから最高の IDE として評価されています．PyCharm は，JetBrains 社が開発した Python 専用の統合開発環境（IDE）です．PyCharm には，コード補完，デバッグ，パッケージ管理（pip），バージョン管理（Git など），統合された開発ツール（クイックナビゲーション，プロファイラなど）の機能があり，大規模なプロジェクトでも効率的に開発できます．そのため，WRS を用いた Python 開発にも適しています．

1　PyCharm のインストールを設定

1.1　無料のコミュニティ版

PyCharm は https://www.jetbrains.com/PyCharm/download からコミュニティ版（Commmunity Edition）を無料で入手できます．プロフェッショナル版（Professional Edition）は有料となります．インストール時に表示されるダイアログボックスにおいて，“add launchers dir to the PATH” のチェックを外してください．インストール後，図 1（右）のようなウィンドウが表示されます．

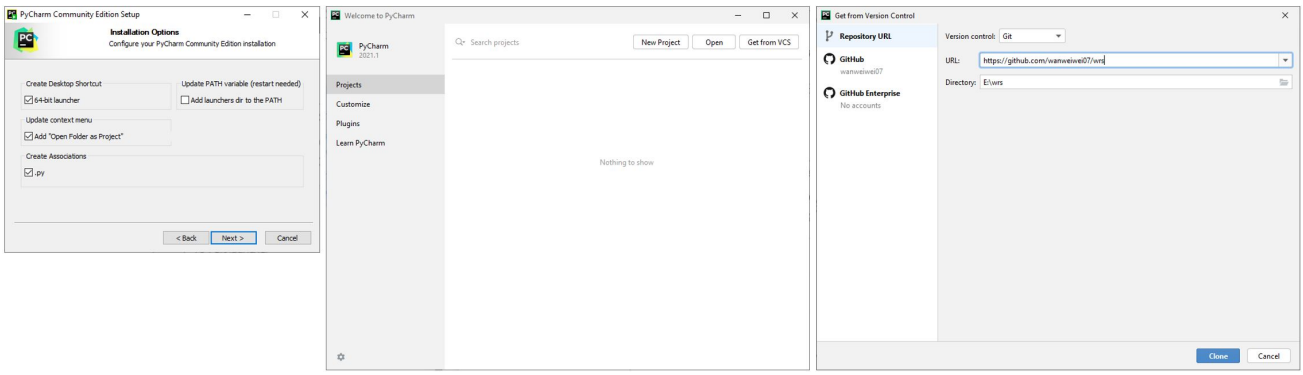

図 1: PyCharm インストール時に表示されるダイアログボックスとその入力方法．左図に示す “add launchers dir to the PATH” のチェックを外してください．中心の図に示す Get from VCS（すでにプロジェクトを開いている場合は Git->Clone）をクリックして，ソースファイルをダウンロードします．インストール後に，右図に示すダイアログボックスが表示されます．

1.2　Git を介して WRS をクローン

WRS のソースコードは https://github.com/wanweiwei07/wrs にて公開しています．ソースコードを個人の PC にインストールする際は，直接ダウンロードではなく，PyCharm の Get from VCS（すでにプロジェクトを開い

ている場合は Git->Clone）によってダウンロードしてください．図 1（右）に示すように，ソースコードの URL（https://github.com/wanweiwei07/wrs）と PC 上の保存フォルダーを設定して，Clone ボタンを押すと，WRS のソースコードは指定されたフォルダーにダウンロードされます．Git がインストールされていない場合，Clone ボタンが灰色に表示され，クリックできません．その際には，赤い文字と指示にしたがって，Git をインストールしてください．ダウンロードが終わると，開発インターフェースが表示されます（図 2）．デフォルト設定では，図 2 左側のプロジェクトのナビゲーションビューが非表示になっています．画面左上のフォルダボタンをクリックすれば開くことができます．また，中心に表示されているプログラムは 0000_example/ur3e_dual_left_planning.py です．任意のファイルをナビゲーションビューにてダブルクリックすれば，このファイル内容も確認できます．上述の方法でダウンロードしたファイルを GitHub のリポジトリ上の最新のものに更新したい場合には，画面の一番上のメニュー欄の main という文字をクリックすることで，Update Project の選択肢が見えます．その選択肢をさらにクリックすると，リポジトリにあるソースコードとの差分だけ自動的に更新されます．差分だけですので，改めて全てをダウンロードする必要はありません．このように，VCS で WRS のソースコードの更新などのバージョン管理が容易になります．

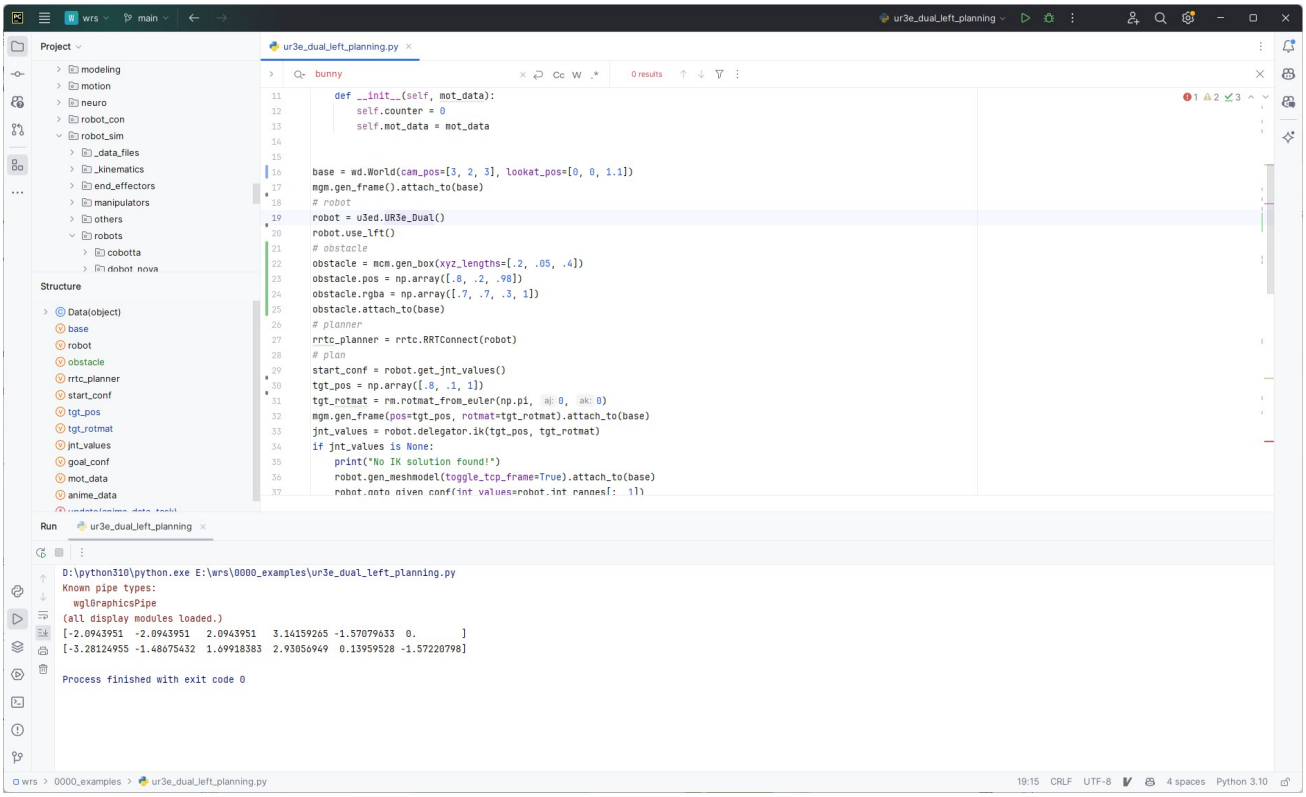

図 2: WRS のソースコードをダウンロードした後の開発インターフェース画面.

1.3 閲覧したファイルやコード間の切り替え

また，画面上部の main の右側に左右反対向きの矢印が 2 つあることに注意してください．この 2 つの矢印は，以前に閲覧したファイルやコードに素早く戻るために使用できます．デフォルトでは，これらの矢印は表示されていません．タイトルバーの空白部分を右クリックし，[Add to Main Menu Bar] -> [Back/Forward] を選択して表示させることができます．また，選択しても表示できない場合，[Customize Toolbar] で手動で [Back/Forward] を探して追加しても構いません．この 2 つの矢印は，多くのファイル間を素早く切り替えるために非常に重要なので，タイトルバーに表示させ，簡単にアクセスできることを強くお勧めします．

2　Python インタープリターの設定

本来，PyCharm をインストールした後，まず Python インタープリターを設定する必要があります．本章では便宜上，Python インタープリターの設定方法を直接紹介せず，必要に応じて設定していきます．

2.1　WRS に使われる Python インタープリターの設定

ここで，ur3_dual_left_planning.py ファイルを実行してみましょう．現時点では Python インタープリターがまだ設定されていないため，ファイルを開くと“No Python interpreter configured for the project”という警告がコードの表示画面の上に表示されます．PyCharm では，プロジェクト用の Python インタープリター，例えば CPython，Jython，あるいは Anaconda のような Python の特定のディストリビューションを設定する必要があります．Python インタプリタを設定するには，[File] -> [Setting] -> [Project] -> [Python Interpreter] の順に進んで設定します．Python Interpreter のの画面のドロップダウンリストで，[Show All...] を選択すると，[Project Interpreters] ウィンドウが表示され，[Project Interpreters] ウィンドウの右上にある+ボタンを押すと，[Add Python Interpreter] ウィンドウが表示されます．[Add Python Interpreter] ウィンドウの左側のリストから [System Interpreter] を選択し，中央に表示されているドロップダウンリストから最新のバージョンを選択すれば Python インタープリタの設定が完了します．もしドロップダウンリストにインストール済みの Python プログラムが見つからない場合は，自分のコンピュータにインストールされている Python のパスを手動で設定することで，Python インタープリターの設定を完了できます．[Add Python Interpreter] ウィンドウを図 3 に示します．Python インタープリターの設定に於いて，[System Interpreter] 以外，Virtualenv，Conda，Pipenv など，他にもいくつかの設定可能な Python 環境の候補があります．初学者は System Interpreter 以外の項目の設定は無視してもよいと思います．

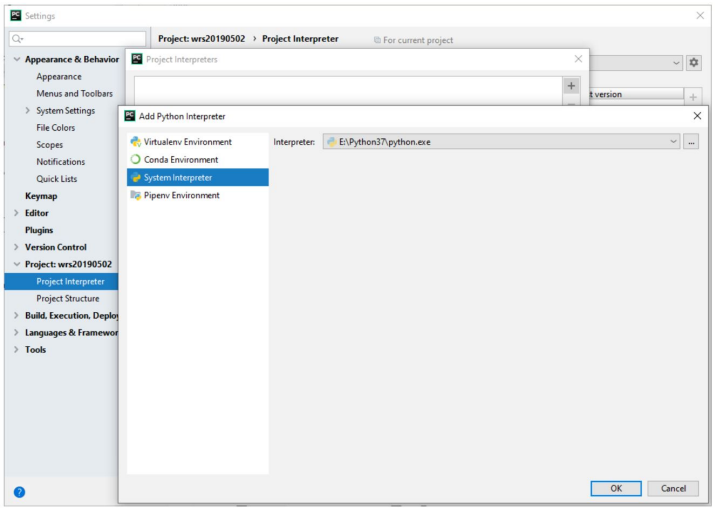

図 3: Python インタプリターの設定画面．

2.2　サードパーティパッケージを追加

2.2.1　必須パッケージ

WRS はいくつかのサードパーティパッケージに依存しています．ur3_dual_left_planning.py を実行する前に，依存パッケージをインストールします．Python インタープリタを設定した後，コマンドラインにて pip（Python から提供されているオリジナルのパッケージマネージャ）を使用するか，設定ウィンドウで+ボタンを押してパッケージを逐一インストールしてください（PyCharm が提供しているユーザーフレンドリなパッケージマネージャ）．設定ウィンドウの+ボタン押下後の画面は図 4 に示しています．ここで，パッケージ管理ツールがないと提示される場合は，画面の指示に従ってインストールをクリックしてください．

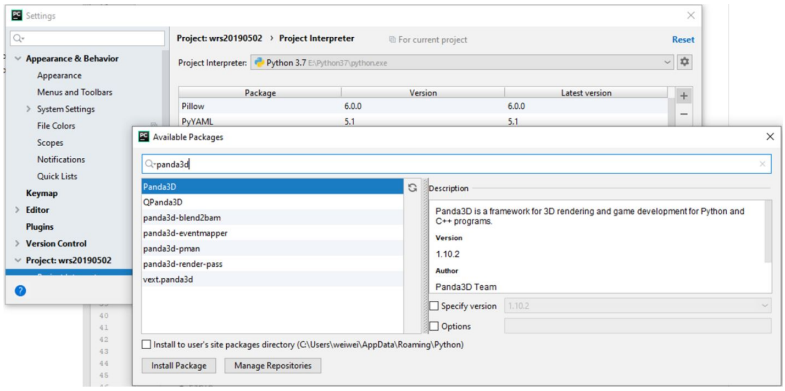

図 4: PyCharm が提供しているユーザーフレンドリなパッケージマネージャの画面表示.

ur3_dual_left_planning.py のプログラムを正常に起動するには，以下のパッケージをインストールする必要があります．これらのパッケージは WRS システムの最小サポートライブラリ要件でもあります.

- ◇ panda3d: ゲームやシミュレーションの開発を支援するオープンソースのゲームエンジンです．グラフィックス，Graphical User Interface (GUI) など，多くの機能を提供しています．WRS システムのグラフィックレンダリングと衝突検出は，Panda3D をベースに開発されています.
- ◇ numpy: 行列計算のためのパッケージです．多次元配列オブジェクトである ndarray を提供し，これを用いて効率的な配列の作成と操作，線形打数の計算などを実行できます．WRS システムの行列計算は numpy に依存します.
- ◇ scipy: 科学技術計算を行うためのライブラリで，numpy に基づいて構築されています．数値微積分，最適化，統計解析などの関数を提供しています．WRS システムの最適化計算は numpy に依存します.
- ◇ scikit-learn: データマイニングやデータ解析のための幅広い機械学習アルゴリズム（分類，回帰，クラスタリング，データの前処理など）を含んでいるパッケージです．WRS システムのクラスタリングや木の操作は scikit-learn に依存します.
- ◇ networkx: グラフ構造を扱うためのライブラリです．複雑なネットワークの作成（有向，無向グラフ），操作（添削，探索），解析を行うためのツールを提供します．WRS システムの経路探索は networkx に依存します.
- ◇ matplotlib: データの可視化を行うためのライブラリです．シンプルなプロットから複雑なグラフや図表まで，様々な種類のグラフを描画するための豊富な機能を提供します.
- ◇ shapely: 幾何学的オブジェクトを操作するためのライブラリです．ジオメトリを簡単に作成，操作，解析するための強力なツールセットを提供します．WRS のメッシュモデルの計算に使われます.
- ◇ pycollada: COLLADA ファイルを読み書きするためのライブラリです．.dae 形式のメッシュモデルの操作に必要とされます.
- ◇ tqdm: プログレスバーを表示するためのライブラリです．tqdm を使用すると，ループの進行状況を視覚的に確認することができ，処理の進行度合いを簡単に把握できます．WRS のデータ駆動計算に利用されます.

以上のパッケージをインストールすると，ur3_dual_left_planning.py を実行できます．開発インターフェースから [Run]->[Run...] をクリックするか，編集領域を右クリックして [Run “xxx”] を選択して実行してみてください．ur3_dual_left_planning.py の実行結果は図 5 に示します．このプログラムの実行結果はアニメーションであり，アニメーションの各フレーム間の更新はユーザーがスペースキーを入力するまで待機します．つまり，ユーザーがスペースキーを一度押すと，アニメーションが一フレーム進みます．確認しやすいように，レンダリングされた実行結果ではロボットと環境の衝突検出のためのバウンディングボックスも描画しました．ロボットの動作生成アルゴリズムは，これらのバウンディングボックス間の衝突を考慮して回避動作を行います．図 5 は，このアニメーションの三つの重要なフレームを示しています．ご覧の通り，ロボットは障害物をうまく回避しています.

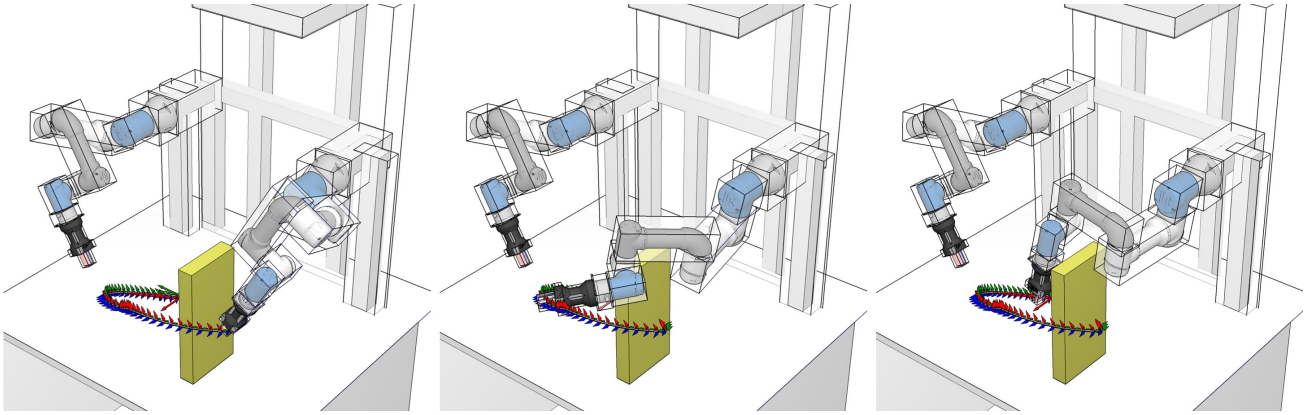

図 5: ur3_dual_planning.py の実行結果.

2.2.2 その他のパッケージ

上記の必須サポートパッケージに加えて，WRS のいくつかの具体的なプログラムでは，以下のライブラリに依存する場合があります.

- ◇ rtree: 高次元データを扱うための更新可能な高速探索木．rtree は通常，空間データに使用されますが，単一の高次元データポイントについては，各データポイントを単点矩形として表現することで実現できます．WRS システムの経路探索に使われることでより大規模かつ効率な動作計画を実装できます.
- ◇ opencv-python: OpenCV の公式 Python バインディングです．画像処理やコンピュータビジョンに必要な主要なモジュール，例えば，画像の読み込みや保存，基本的な画像変換，フィルタリング，エッジ検出，カメラキャリブレーションなどが含まれています.
- ◇ opencv-contrib-python: OpenCV の ”contrib” モジュール（拡張機能）を含むパッケージです．opencv-python には含まれていない実験的な機能や追加のアルゴリズムが含まれています．SIFT（Scale-Invariant Feature Transform），SURF（Speeded-Up Robust Features），そして様々なオブジェクト検出やトラッキングのアルゴリズムが挙げられます．黒白マーカに必要となる aruco モジュールは，OpenCV の contrib モジュールの一部として提供されています.
- ◇ open3d: 3D データの処理，操作，および可視化のためのオープンソースライブラリです．点群，メッシュ，RGB-D 画像などの 3D データを扱うための高度な機能を提供します.
- ◇ mujoco: 高性能な物理シミュレーションエンジンであり，ロボティクスや生物学的モデリング，機械学習の研究に広く使用されています．Gymnasium（旧 OpenAI Gym）や Isaac Gym（Omniverse）などロボット学習用シミュレータに採用されています．WRS は mujoco のレンダリングエンジンとして使われるのは可能です.
- ◇ grpcio, grpcio-tools: Google が開発した高性能なオープンソースの RPC（Remote Procedure Call）フレームワークです．gRPC を使用することで，異なるマシンや異なる言語で実装されたサービス間で効率的に通信を行うことができます．gRPC は，HTTP/2 を使用し，プロトコルバッファ（Protocol Buffers）をデフォルトのシリアライゼーションフォーマットとして使用します.

3 力試し

この小節では，PyCharm を使って独自の Python プログラムを作成する方法を試みます．WRS のライブラリや関数を便利に使用するために，クローンした WRS プロジェクトのフォルダー内にプログラムを作成します．WRS フォルダーには既に多くのフォルダーやファイルがあります．既存の内容と区別しやすくするために，新しいフォルダー 000000_study を作成し，このフォルダー内にプログラムを作成していきましょう．この名前はおかしいと思われるかもしれませんが，特別な意味はなく，先頭の 000000 はこのフォルダーを PyCharm のナビゲーターの上位に表示させるためのものです.

3.1 オブジェクト指向プログラミング

フォルダーを作った後，フォルダーの下に list2.3-1-class.py という新しいファイルを作成し，第一章で紹介したオブジェクト指向プログラミングの例を試してみます．list2.3-1-class.py の中身は下記のリスト 2.3-1 に示します．

リスト 2.3-1: オブジェクト指向プログラミング

```
    class Car(object):
        def __init__(self, brand, speed):
            self.brand = brand
            self.speed = speed

        def accelerate(self):
            self.speed += 10
            print(f"The car is now going at {self.speed} km/h")

    class ElectricCar(Car):
        def __init__(self, brand, speed, battery):
            super().__init__(brand, speed)  # 親クラスの初期化を呼び出す
            self.battery = battery  # 新しい属性を追加

        def charge(self):
            print(f"The battery is now charged to {self.battery} kWh")

    if __name__ == '__main__':
        my_electric_car = ElectricCar("Tesla", 70, 100)
        my_electric_car.accelerate()
        my_electric_car.charge()
```

右クリックして [Run] を選択してこのファイルを実行すると，PyCharm の下の実行結果のタブに以下のような結果が表示されます．PyCharm は E ディスクの python311 フォルダー内の python.exe インタープリターを使って E ディスクの wrs フォルダー内の 000000_study の下の list2-1-class.py ファイルを解釈したことが分ります．実行結果は第一章のコマンドライン下の結果と同じです．

```
D:\python311\python.exe E:\wrs\000000_study\list2-1-class.py
The car is now going at 80 km/h
The battery is now charged to 100 kWh

Process finished with exit code 0
```

このプログラムは，第一章のコマンドライン入力といくつかの細かな違いがあります．ここではそれについて簡単に説明します．まずは，Car も object クラスから継承することです．Python のクラスが object を継承することが推奨されます．object を継承することで，一貫した基底クラスを持つ，プロパティ，super() 関数などの高度な機能も利用できます．list2-1-class.py では，親のない新しい Car クラスを宣言する際，object から継承するようにしています．次は，18 行の if ___name___ == '___main___' は，ファイルが直接実行された場合にのみ特定のコードを実行するための便利な方法です．___name___は Python の特別な組み込み変数で，ファイルがどのように実行されているかを示します．list2-1-class.py を直接実行すると，___name___の値は'___main___' になります．if 文の後ろの部分が実行されます．list2-1-class.py を他のモジュールからインポートした場合，___name___の値はそのモジュール名になります．その時，___name___ == '___main___' は偽になり，if 文の後ろの内容は実行されません．もちろん，現在のファイルが他のコードによってモジュールとして使用されない場合，この条件文を使用せずに，実行したいコードを直接書いても構いません．

3.2 三次元描画

次に，三次元画面を描画してみましょう．000000_study フォルダーの下に list2-2-coordinate.py という新しいファイルを作成し，下記のコードを入力して実行してみてください．

```
    import numpy as np
    import modeling.geometric_model as mgm
    import visualization.panda.world as wd

```

```
5     base = wd.World(cam_pos=np.array([1, .8, .6]), lookat_pos=np.zeros(3))
6     frame_model = mgm.gen_frame()
7     frame_model.attach_to(base)
8     base.run()
```

このファイルは, サードパーティの numpy パッケージを import することに加えて,wrs システムの modeling フォルダ内の geometric_model ファイルと,visualization フォルダ内の panda フォルダにある world ファイルも import しています．list2-2-coordinate.py の 6 行目では,world ファイルで宣言された World クラスを用いて base オブジェクトを定義しています．7 行目では,geometric_model ファイルで定義された gen_frame() 関数を用いて frame_model を作成します．8 行目では, 作成した frame_model を base にくっ付けます．9 行目では,base を実行して, くっ付けられた三次元モデルを三次元画面に描画します．図 6 の左側には描画の結果を示します．描画されたウィンドウ内でマウスの左ボタンをクリックしながらドラッグしたり，ホイールを回転させたりすることで，三次元画面の視角やズームを変更できます．三次元画面の初期視角とズームは，第 6 行で base オブジェクトを定義する際に渡されたパラメータによって決まります．一番目の引数は，初期カメラの位置を x 軸正方向 1m，y 軸正方向 0.8m，z 軸正方向 0.6m に設定されています．二番目の引数は，カメラを原点 x=0m，y=0m，z=0m を向かせています．図 6 の右側には設定したカメラの様子を示しています．半透明の灰色の物体はカメラです．カメラの前面に映った画像が描画されています．この画像も WRS システムで作ったプログラムの実行結果でした．詳しいコードはリスト 2.3-2 に参考してください．

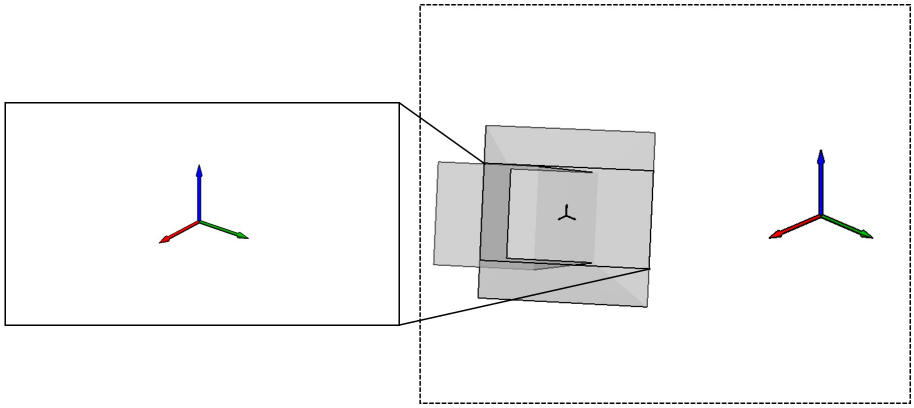

図 6: base オブジェクトを定義する際に渡されたパラメータによって設定したカメラの様子.

リスト 2.3-2: カメラの様子を示すプログラム

```
1     import numpy as np
2     import modeling.geometric_model as mgm
3     import visualization.panda.world as wd
4     import visualization.panda.panda3d_utils as pdu
5
6     # 実際のカメラのパラメータ
7     cam_pos = np.array([1, .8, .6])
8     lookat_pos = np.zeros(3)
9     # このプログラムのための仮想カメラのパラメータ
10    cam_pos_for_rendering = np.array([1.5, 1.5, 1])
11    lookat_pos_for_rendering = np.array([0, 0, 0])
12    # このプログラムのためのワールド
13    base = wd.World(cam_pos=cam_pos_for_rendering, lookat_pos=lookat_pos_for_rendering)
14    # フレームの描画
15    frame_model = mgm.gen_frame()
16    frame_model.attach_to(base)
17    # 仮想カメラの描画
18    cam_res = np.array([1920, 1080])
19    screen_size = np.array([0.096, 0.054])
20    v_cam = pdu.VirtualCamera(cam_pos, lookat_pos, resolution=cam_res,
21                              screen_size=screen_size)
22    v_cam.gen_meshmodel().attach_to(base)
23    display = pdu.Display("virtual_cam_display", v_cam.screen_size)
24    display.attach_to(v_cam)
```

```
25      base.run()
```

4 PyCharmを効率的に使うためのポイント

最後に，PyCharm を効率的に使うためのコツを整理しておきます.

- ◇ 閲覧したファイルやコード間の切り替え: 小節 1.3 に説明した通り，移動元に戻るのは，タイトルバーの左側の [Back/Forward] ボタンを使ってください．[Back/Forward] ボタンが表示されない場合，タイトルバーの空白部分を右クリックで追加してください.
- ◇ 記述されている関数の中身への移動: 記述されている関数の中身に移動するためには，ctrl キーを押しながら関数の名前をクリックしてください．関数の中身から移動元に戻るのは [Back/Forward] ボタンを活用してください.
- ◇ 呼び出したい関数の引数の確認: 関数のパラメータを見るには，パラメータリスト内のどこかにカーソルを移動し，ctrl+p を押してください．そして，画面上にパラメータの名前や型など一行に表示されます.
- ◇ ファイルの実行: Python ファイルを実行するには，開発インターフェースのメニューで [Run]->[Run...] をクリックするか，タイトルバーの右の緑色の三角ボタンを押すか，または編集領域で右クリックして [Run “xxx”] をクリックしてください．実行予定のファイルに引数を渡したい場合，タイトルバーの右の実行名のドロップダウンリストをクリックし，[Edit Configurations] を介して実行コンフィギュレーションを設定してください．ただし，多くの実行コンフィギュレーションを残さないように注意してください．実行コンフィギュレーションの確認および削除は [Run]->[Edit configurations] をクリックしてください．デフォルトの設定では，ほとんどのスクリプトが実行可能です．作成した設定をすべて削除すればデフォルトの設定に戻ります.
- ◇ 書式の自動整理: ctrl+alt+l を同時に押すと，乱れた空白や改行が自動的に添削され，プログラムの書式も整理されます.
- ◇ 同一内容の快速修正: alt+j を同時に押すと，現在選択している文字列を後ろのプログラムから検索し，同じものも選択されます．複数回を押すと，されに後ろのものも選択されます．同時に選択された文字列を修正すると，全員に反映されます．alt+shift+j は alt+j の逆命令で，これらのキーの組み合わせを同時に押すと新しく選択された文字列が，選択された順序に従って新しいものから古いものまで削除されます.
- ◇ 複数の列と行の領域に同時に追加・削除: alt+shift+左クリックを押しながらドラッグすることで，複数の列や行を跨いでファイルの領域を選択でき，それらの選択領域に同時に文字を追加・削除することができます.
- ◇ 文字列の快速選択 ctrl+w を繰り返し押すと，テキスト選択範囲が以下の順序で細かい選択から大きな選択まで調整することができます．1）カーソルが置かれている単語全体を選択．2）その単語を含む文全体を選択．3）さらにその文を含む段落全体を選択．4）最終的には，ファイル全体を選択．細かい選択から大きな選択までスムーズに行うことができます.