Анимация в среде *Fusion 360* – дорогами *API*

А.Ю. Стремнев, к.т.н. (БГТУ им. В.Г. Шухова) *nml235l@yandex.ru*

Использование компьютера для автоматизации проектных работ “балует” конструктора обилием таких возможностей, которые до недавних пор можно было получить, пожалуй, только на натурных образцах-моделях. В первую очередь это касается сборок, содержащих как отдельные подвижные элементы, так и целые кинематические схемы. Проектный анализ таких сборок без воплощения электронных прототипов в материальную форму позволяет экономить значительные средства. Действительно, очень удобно проверить функционирование изделия, предварительно оценив “работоспособность” его модели на экране монитора.

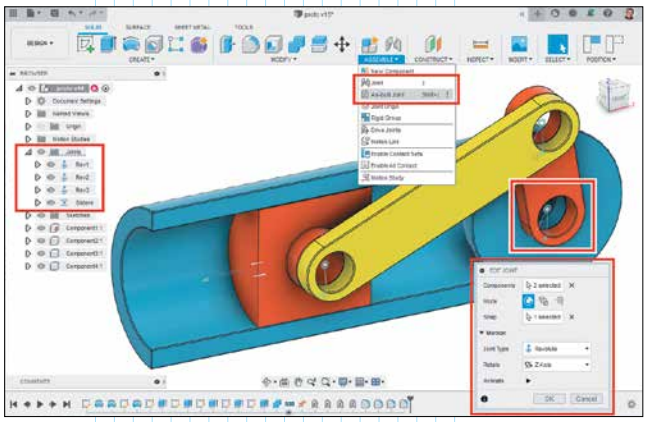

Рис. 1

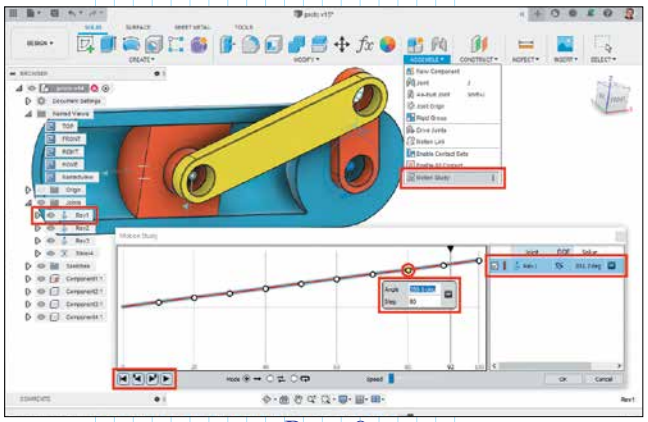

Рис. 2

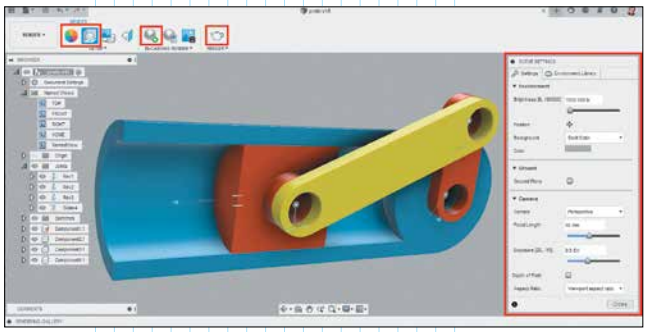

Рис. 3

Посмотрим, как обстоит дело с тестированием виртуальных моделей сборок в системе *Autodesk Fusion 360*.

Наша модель – инженерная классика – кривошипно-шатунный механизм (рис. 1). В этой сборке имеются типовые кинематические пары (три вращательные и одна поступательная), а каждая из подвижных деталей (кривошип, шатун и поршень) совершает по-своему уникальное движение. В среде *Fusion 360* связать детали кинематическими парами позволяет команда ***Joint*** (Соединение) или ***As-build Joint*** (Соединение по месту) из группы команд ***Assemble*** (Сборка). При создании соединения требуется указать ключевые элементы на связываемых деталях и выбрать тип кинематической пары. Например, в нашем случае кривошип будет соединен с цилиндром связью типа ***Revolve*** (Вращение). Именно она после реализации остальных соединений будет приводить механизм в действие. Отметим, что все добавленные соединения получают уникальные имена и доступны для редактирования в браузере модели.

Самый простой способ проверить модель в действии заключается в “ручной” буксировке какого-либо подвижного элемента – тогда связанная с ним кинематическая схема придет в движение. Другой вариант – заставить модель двигаться самостоятельно. Для этого *Fusion 360* предлагает инструмент ***Motion Study*** (Анализ движения) из той же группы *Assemble*.

Работа с ним заключается в следующем. После выбора соединения, подвижность которого нужно проверить, на шкале времени следует задать точки со значениями варьируемого параметра этого соединения (рис. 2). В исследуемой модели таким параметром для ключевого соединения – вращательной пары между кривошипом и цилиндром – является угол поворота. Кнопки “проигрывателя” в нижней части окна *Motion Study* позволяют запустить анимацию выбранного соединения, а вместе с ним привести в движение всю кинематическую схему. При необходимости в этом окне можно указать изменение по времени для нескольких соединений и проверить работу механизма при одновременном изменении параметров соответствующих кинематических пар.

Инструментарий *Motion Study* весьма лаконичен – например, нет возможности записать движение механизма в видеофайл, а шкала времени проградуирована в условных процентах от общей продолжительности варьирования параметра. И, наконец, кинематика сборки отображается в рабочем поле в далеко не фотореалистичном виде. Для отладки модели это допустимо, но для маркетинговой презентации готового прототипа понадобится более качественная визуализация.

Autodesk Fusion 360 предлагает отдельную среду для фотореалистичного представления модели: набор команд ***Render***, включающий в себя богатую палитру для работы с материалами, и редактор сцены (рис. 3). Сам процесс визуализации

(рендеринга) можно выполнять как на рабочем месте пользователя (*In-Canvas*), так и задействовав облачный ресурс *Autodesk*.

После запуска визуализации и по достижению необходимого качества готовое изображение модели

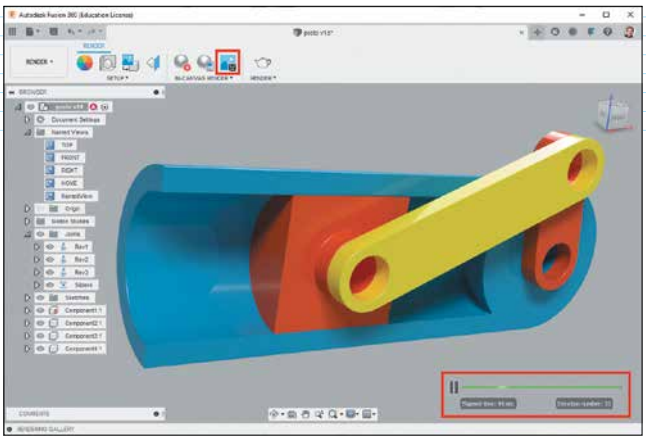

Рис. 4

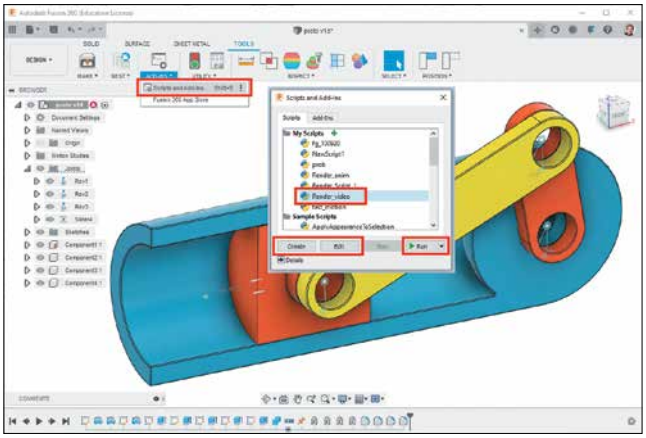

Рис. 5

```
Render_video.py
1  import adsk.core, adsk.fusion, adsk.cam, traceback
2  import math
3  import time
4  app = adsk.core.Application.get()
5  design = adsk.fusion.Design.cast(app.activeProduct)
6  root = design.rootComponent
7  def run(context):
8      ui  = app.userInterface
9      revs = 1
10     StepValue = 36
11     stepNo = 0
12     driveJoint = root.asBuiltJoints.itemByName('Rev1')
13     revMotion = adsk.fusion.RevoluteJointMotion.cast(driveJoint.jointMotion)
14     renderWS = ui.workspaces.itemById("FusionRenderEnvironment")
15     renderWS.activate()
16     adsk.doEvents()
17     for i in range(revs):
18         for step in range(0,360,StepValue):
19             angle = (math.pi / 180) * step
20             revMotion.rotationValue = angle
21             adsk.doEvents()
22             app.activeViewport.refresh()
23             adsk.doEvents()
24             cmdDefs :adsk.core.CommandDefinitions = ui.commandDefinitions
25             cmdDef  :adsk.core.CommandDefinition = cmdDefs.itemById('InCanvasRenderCommand')
26             cmdDef.execute()
27             for i in range(500):
28                adsk.doEvents()
29             imageFolder = 'D:/My/Fusion_anim/sequence/'
30             frameNumber = stepNo
31             filename = imageFolder + "Test-" + str(frameNumber).zfill(4) + '.jpg'
32             app.activeViewport.saveAsImageFile(filename, 0, 0)
33             adsk.doEvents()
34             cmdDef.execute()
35             adsk.doEvents()
36             stepNo += 1
37     ui.messageBox('Finished')
```

Рис. 6

можно сохранить в графическом файле (рис. 4). На первый взгляд всё очень удобно, но как же быть с анимацией сборки? В *Motion Study* движение в модели воспроизводилось, но его отображение было далеко не идеальным, и сохранить его было нельзя. В среде же *Render* всё обстоит практически наоборот: есть весьма достойная визуализация, а вот поддержка движения не предусмотрена. Таким образом, задача получения качественной анимационной последовательности средствами *Fusion 360* кажется нерешаемой.

Но, вместе с тем, среда *Render* содержит практически всё необходимое: есть и команда генерации изображения, и средство его сохранения. Если мы будем "вручную проворачивать" механизм, лежащий в основе сборки, а каждое новое положение визуализировать в среде *Render*, то на выходе может получиться качественная "раскадровка" анимации, которую без труда "соберет" практически любой видеоредактор. Итак, алгоритм есть, необходимо определиться с тем, кто (или что) его будет выполнять.

Рутинные, часто повторяющиеся действия лучше всего поручить компьютеру, поэтому обратимся к скриптовым возможностям *Fusion 360*. Для этого выберем на вкладке ***Tools*** (Инструменты) команду ***Scripts and Add-Ins*** (Скрипты и дополнения) и, нажав кнопку ***Create*** (Создать), напишем программу-скрипт на языке *Python* (рис. 5).

В коде скрипта (рис. 6), после инициализации приложения *Fusion 360* (***app***) и текущей сборки (***root***), следует задать значения переменных ***revs*** и ***StepValue***, определяющих количество полных оборотов кривошипа для анимации и шаговый угол поворота кривошипа между отдельными визуализируемыми кадрами соответственно. Далее получаем доступ (по имени) к вращательной паре-соединению между кривошипом и цилиндром (см. рис. 1) строкой:

```
driveJoint=root.asBuiltJoints.itemByName('Rev1')
```

К возможности изменения угла поворота в этом соединении обращаемся с помощью команды:

```
revMotion=adsk.fusion.RevoluteJointMotion.
cast(driveJoint.jointMotion)
```

После этого вызываем среду визуализации *Fusion 360* (*"FusionRenderEnvironment"*) и начинаем программно вращать кривошип с помощью цикла ***for*** на величину шага (*StepValue*).

На каждом шаге устанавливаем текущее положение кривошипа (строка ***revMotion.rotationValue=angle)*** и запускаем локальную визуализацию (команда ***'InCanvasRenderComand').***

При этом следует дать системе *Fusion 360* время (порядка 40 секунд) на генерацию изображения для текущего положения кривошипно-шатунного механизма, что осуществляется с помощью пустого цикла:

```
for i in range(4000): adsk.doEvents()
```

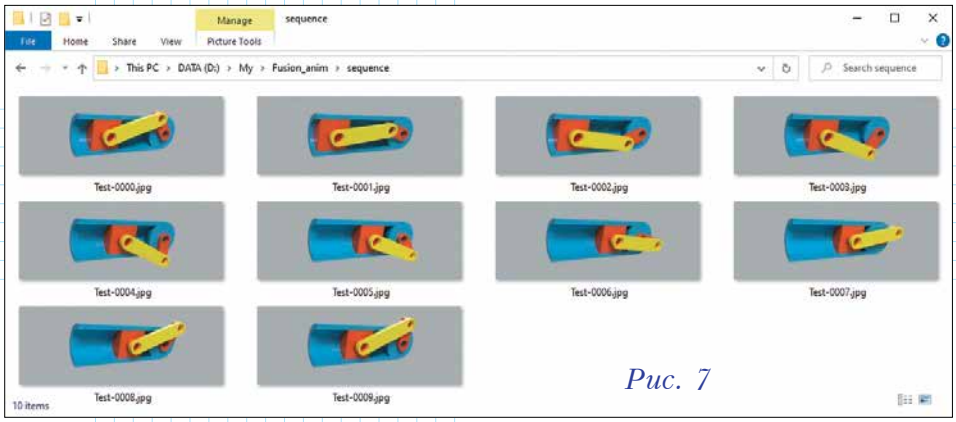

Рис. 7

Наконец, после визуализации каждого кадра его необходимо сохранить в виде графического файла инструкцией:

app.activeViewport.saveImageFile(filename,0,0)

Имя каждого следующего кадра (***filename***) включает в себя порядковый номер, получаемый на основе инкремента счетчика цикла (***stepNo+=1***).

Готовый скрипт запускаем в окне ***Scripts and Add-Ins*** командой ***Run*** (см. рис. 5) и дожидаемся его завершения. В ходе работы скрипта производится поворот кривошипа с заданным шагом, визуализация изменившегося состояния модели сборки и сохранение готовых изображений-кадров в графические файлы (рис. 7).

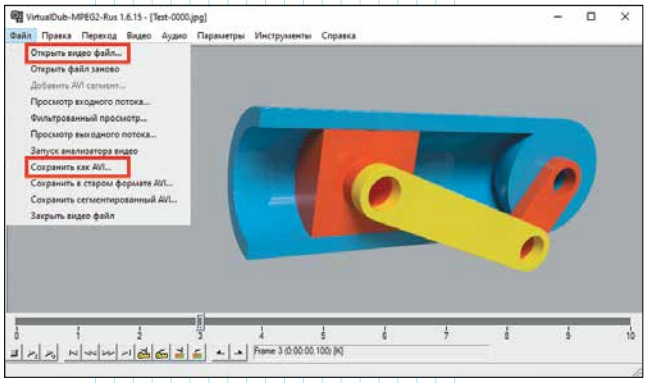

Рис. 8

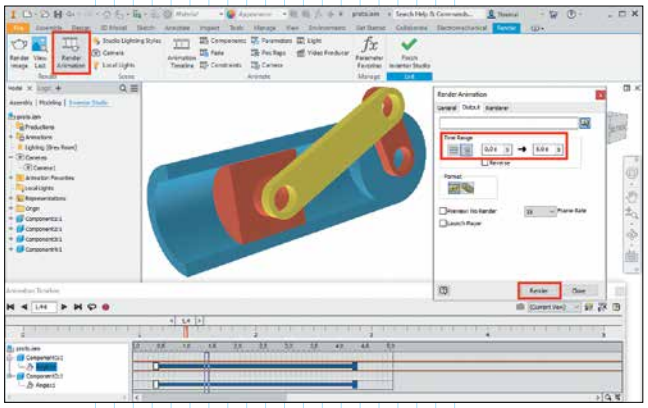

Рис. 9

Поскольку имена файлов-кадров на выходе из скрипта получаются пронумерованными, то видеоредактор (в нашем случае это программа ***VirtualDub***) автоматически загрузит их на свою шкалу времени (Файл → Открыть видео файл), после чего анимационную последовательность можно записать в один общий видеофайл командой "Сохранить как *AVI*" (рис. 8).

Готовое видео анимированной сборки будет иметь тем большую плавность, чем меньше величина шага изменения варьируемого параметра соединения (***StepValue***), для каждого положения которого предлагаемый скрипт генерировал и сохранял кадры.

Итак, поставленная задача решена: получен видеофайл фотореалистичной анимированной модели сборки. Впрочем, рекомендовать этот путь начинающим пользователям универсальной *CAD*-системы *Fusion 360* мы, пожалуй, не будем. Следует немного подождать и, скорее всего, разработчики из компании *Autodesk* добавят функционал шкалы времени в набор команд *Render*. Такой опыт у них уже есть – это реализовано в "старшем брате" системы *Fusion 360*, известном под именем *Autodesk Inventor*, в среде ***Inventor Studio*** (рис. 9).

Желающие же опробовать упоминаемый скриптовый код могут найти его по ссылке [1], попутно расширив свои знания о структуре *Fusion 360 API* [2].

Наглядное представление результатов автоматизированного проектирования в виде анимации моделируемого изделия является хорошим дополнением к конструкторской документации, особенно в целях маркетингового продвижения. В арсенале системы *Fusion 360* имеется ряд инструментов для варьирования пространственного положения компонент сборок и фотореалистичной визуализации. Использование интерфейса прикладного программирования (*API*) позволяет аккумулировать эти возможности для подготовки покадровой анимационной последовательности.

Полезные ссылки:

1. Архив рассматриваемого проекта (*Fusion 360*) + скрипт (*Python*) // https://disk.yandex.ru/d/i_73oUVYcB-jmw

2. Онлайн-справка по *Fusion 360 API* // https://help.autodesk.com/view/fusion360/ENU/?guid=GUID-A92A4B10-3781-4925-94C6-47DA85A4F65A

Об авторе

Александр Юрьевич Стремнев – кандидат технических наук, доцент кафедры информационных технологий Белгородского государственного технологического университета им. В.Г. Шухова