ChatGPT

Примеры 3D-симуляции сжатия бумаги (скомкивания)

Ниже приведены **десять публичных примеров** проектов с открытым исходным кодом, реализующих 3D-анимацию скомкивания или сжатия бумажного листа. Для каждого указано название или краткое описание проекта, ссылка на репозиторий или ресурс, используемые технологии, способ реализации сминания (cloth-система, soft body и т.п.), возможность наложения текстуры на модель, а также сведения о запуске (версия Blender или запуск на Kaggle и т.д.). Каждый пример сопровождается кратким описанием и упоминанием наличия визуальных результатов.

1. Cloth-симуляция в Blender с коллизией (сфера сжимает лист)

Проект: Метод скомкивания листа в Blender с помощью cloth-симуляции и анимированного коллайдера (описан, например, в туториале CGMatter).
Репозиторий/Ресурс: Видео-урок CGMatter (сводка на Lesterbanks) [1].
Технологии: Blender (Physics: Cloth), модификатор **Cloth** и объекты-коллайдеры.
Реализация скомкивания: Плоскому объекту (листу бумаги) назначается **Cloth**-симуляция со свойствами, имитирующими жесткость бумаги. В сцене анимируется **сфера-коллайдер**, которая давит на лист со всех сторон, сминая его. При правильных настройках ткань собирается в комок под воздействием движущейся сферы [1]. Этот метод позволяет **анимировать процесс сжатия** – бумага постепенно морщится и собирается вокруг сферы.
Текстурирование: Да, на плоскость можно наложить изображение как UV-текстуру перед симуляцией. В уроке демонстрируется применение текстуры бумаги на лист [1], и после симуляции смятая модель сохраняет текстурные координаты (поэтому видны складки на текстуре).
Запуск: Требуется **Blender** (подходят версии 2.8 и выше). Настройки cloth (например, высокое напряжение, плотность как у бумаги) и анимация сферы выполняются в интерфейсе Blender. Для воспроизведения примера нужно открыть .blend-файл или самостоятельно настроить сцену по инструкции. Можно запускать и через Blender в режиме фона с Python-скриптом, но в данном случае обычно используются интерактивные настройки.
Визуальный результат: Да, присутствует. Итог – реалистично **скомканный шарик из бумаги**. В видео-уроке приводится анимация процесса сжимания. Метод даёт достоверные морщины и складки, хотя требует подбора параметров (масса вертексов, сгибаемость и т.д.) для имитации жесткой бумаги.

Источник: CGMatter tutorial (Blender cloth + collision sphere) [1]

2. Cloth-симуляция в Blender с минимальной гибкостью (нулевая упругость)

Проект: Использование свойств **Cloth** в Blender с близкими к нулю параметрами жесткости, чтобы ткань вела себя как мнущаяся бумага. (Упоминается, например, Gleb Alexandrov в “Blender

Destruction Tutorial").
Репозиторий/Ресурс: Статья *CreativeShrimp – "9 Ways to Destroy Things"*, раздел про cloth [2].
Технологии: Blender (Cloth Physics).
Реализация скомкивания: В данном подходе плоскость также становится Cloth-объектом, но все коэффициенты **жесткости (Stretch, Bend)** устанавливаются очень низкими (практически 0). Такая "ткань" лишена упругости и легко мнется. При столкновении с препятствиями или под действием гравитации лист складывается в хаотичный комок. **Cloth-система с нулевой жесткостью** эффективно симулирует сминание и деформацию поверхности [2]. Для запуска можно, например, "уронить" лист на сцену с коллайдерами – из-за отсутствия жесткости он сам сморщится. Этот метод не требует внешнего анимированного объекта – достаточно силы тяжести и столкновений.
Текстурирование: Возможно – как и в предыдущем примере, объект имеет UV-развертку, поэтому можно применять изображение (например, текстуру страницы) до симуляции. При смятии текстура будет следовать геометрии (однако возможны растяжения текстур, так как mesh деформируется).
Запуск: Достаточно **Blender 2.8+**. Настройки: включить Cloth для плоскости, выбрать предустановку материала с очень высокой жесткостью (или вручную: Tension, Compression, Shear ~ 0, и/или Leather/Denim пресеты), убрать сопротивление изгибу. Можно задать **Gravity** и включить **Self-Collision**. При старте симуляции лист сам сомнется под своим весом или при взаимодействии с окружением. Для ускорения расчета нужны достаточно плотная сетка и мелкий шаг симуляции, что нагружает CPU.
Визуальный результат: Да, получается **смятая бумага с хаотичными складками**. Этот метод отмечен как один из способов "разрушения" в Blender [2]. Он хорошо передает мелкие морщины, но в реальности бумага сохраняет складки (пластические деформации), чего базовая cloth-симуляция Blender не учитывает – результирующая форма похожа, но слегка "резиновая". Однако визуально, особенно с корректной коллизией, эффект убедителен.

Источник: CreativeShrimp (Cloth sim + 0 stiffness for crumpling) [2]

3. Soft Body-симуляция в Blender (мягкое тело вместо cloth)

Проект: Пример использования **Soft Body** физики Blender для имитации динамики бумаги (применялось для анимации летающих бумажек, пакетов и т.п. – например, в работах Ian Hubert).
Репозиторий/Ресурс: Упоминание на Blender.StackExchange о технике Ian Hubert [3].
Технологии: Blender (Physics: Soft Body).
Реализация скомкивания: В режиме Soft Body плоскость воспринимается как объект с вершинами, соединенными пружинками. Для имитации бумаги настраивают высокое сопротивление растяжению вершин (чтобы длина ребер почти не менялась, ведь бумага не тянется) и умеренное сопротивление сгибу. **Soft Body** может обеспечить, что площадь листа сохраняется, а сгибы получаются острыми (при правильной настройке пружин) [4] [5]. Например, Ian Hubert **анимировал летящие листы бумаги** с помощью Soft Body, чтобы они естественно трепетали и морщились в воздухе [3]. При столкновениях или специальных силах лист также может комкаться. В отличие от cloth-модификатора, Soft Body иногда дает более жесткие складки (ближе к поведению бумаги), хотя сложнее в настройке коллизий. Можно создавать **анимацию сдавливания**: например, поместить мягкотелый лист между двумя сближающимися коллайдерами – лист будет морщиться и складываться.
Текстурирование: Полностью поддерживается – soft body не влияет на UV. Можно применять любой материал или изображение на плоскость до симуляции. После смятия текстура "натянута" на получившуюся форму, как на листе бумаги.
Запуск: Blender 2.8+ (или более старые версии, т.к. Soft Body давно в Blender). Нужно перейти в свойства физики объекта и включить **Soft Body**, отключить цель (Goal) или настроить весовые

коэффициенты вершин, задать Edge Spring = 1 (жесткие связи), настроить коэффициенты *Plasticity* (если нужна остаточная деформация), и включить **Self-Collision**. Часто требуется увеличить итерации солвера, шаг по времени и т.д. Запуск на Kaggle возможен через Blender с CLI, но придётся скриптом настраивать параметры.
Визуальный результат: Есть. При правильно заданных параметрах лист удерживает форму и образует **острые заломы**, характерные для бумаги (в отличие от cloth-симуляции, где края могут оставаться гладкими). Такая техника применялась в производстве – например, для реалистичного **полета и помятости пакетов/бумаг** (см. работы Ian Hubert) [3]. Результат – сочетание мягкой динамики и жёстких складок. Однако настройка сложна: по отзывам, нужно балансировать массу вершин и жесткость пружин, иначе лист или "провалится" сквозь объекты, или будет взрываться.

Источник: BlenderArtists/StackExchange (использование Soft Body для бумаги) [3]

4. Скриптовая cloth-симуляция в Blender (автоматизация, пример BinRoot)

Проект: Blender-Cloth-Simulation от пользователя BinRoot – скрипты на Python для Blender, автоматически складывающие ткань (применялось для экспериментов с роботами, складывание одежды) [6] [7].
Репозиторий: BinRoot/Blender-Cloth-Simulation (архивированный) [8] [7].
Технологии: Blender Python API, физический движок Blender (Cloth).
Реализация скомкивания: Скрипты генерируют список случайных точек на сетке ткани, затем последовательно **поднимают и отпускают** ткань за эти точки, заставляя её **сминаться**. По сути, cloth-симуляция управляется Python-скриптом: он фиксирует (pin) определенные вершины, поднимает их, затем отпускает, позволяя ткани упасть и сложиться хаотично [6] [7]. Повторяя процесс с разными точками, ткань (или лист бумаги) мнется всё сильнее. Итог – сильно скомканная форма. Скрипт сохраняет **каждый полученный меш** (состояние ткани) в папку, можно выбрать нужную степень смятости [7]. Хотя изначально проект нацелен на складывание ткани, те же принципы применимы к бумаге (просто задать более жесткие параметры ткани).
Текстурирование: Да, изначальный объект – обычный меш, можно назначить UV и текстуру. Blender сохранит текстурные координаты, а скрипт выводит результирующие меши. Поэтому возможно наложение изображения на полученный скомканный объект (например, если имитируется скомканный плакат – текстура останется видимой).
Запуск: Требуется установленный **Blender** (проект тестировался на Blender 2.79/2.8). Для запуска: поместить модель листа (например, `.3ds` файл плоскости) и выполнить команду в терминале Blender: `blender --background --python gen_points.py <file.3ds>` для генерации точек [9], затем `python cloth_simulation/run.py` для самой симуляции [10]. Скрипт работает и без GUI (фоновый режим), что допускает запуск в среде Kaggle (при установке Blender). Результаты (объекты каждой итерации) сохраняются, их можно импортировать или визуализировать.
Визуальный результат: Да. В репозитории есть изображения: ткань, подвешенная за точку, и сложенная под своим весом [8]. После серии таких операций получается **сильно помятая ткань/бумага с реалистичными складками**. Этот пример демонстрирует полный **pipeline автоматического "комкания"**: от скрипта генерации до сохранения финальной геометрии [7].

Источник: README проекта BinRoot/Blender-Cloth-Simulation [6] [7]

5. SoftGym – симуляция смятой ткани на базе Nvidia FleX

Проект: SoftGym – набор сред для обучения агентов манипулировать мягкими объектами (в том числе тканью) [11]. В SoftGym есть сцены, где полотно изначально **скомкано** и затем расправляется, что подразумевает умение *симулировать смятие*.
Репозиторий: Xingyu-Lin/SoftGym [11] [12].
Технологии: Python, физический движок **NVIDIA FleX** (через обертку PyFlex) [13], OpenGL рендеринг. Работает на GPU (CUDA).
Реализация скомкивания: SoftGym непосредственно содержит окружения *FoldCrumpledCloth* и *SpreadCloth*, где ткань стартует в **скомканном состоянии** [14]. Смятие создаётся процедурно: сначала ткань подвергается случайным силам и столкновениям (например, бросается и переминается виртуальным "руками" в симуляции), итоговое состояние фиксируется как начальное для задачи расправления [12] [14]. Таким образом, SoftGym предоставляет **генерацию реалистично помятой ткани** – с множеством складок и заломов. FleX использует частицы и пружины, обеспечивая правдоподобную физику тонкого листа, включая самоколлизии. Например, в задаче *FoldCrumpledCloth* агент видит перед собой **в комок смятую ткань** и должен её развернуть [14] – то есть движок уже просчитал процесс смятия. Разработчики отмечают, что симуляция достаточно реалистична и близка к реальным данным по поведению ткани [15].
Текстурирование: В SoftGym акцент на физике, визуально ткань однотонная. Однако, модель ткани – это сетка, для которой можно задать текстуру (на этапе рендера OpenGL). С технической точки зрения, FleX оперирует вершинами и треугольниками, поэтому при желании можно привязать UV-координаты и отрендерить с изображением (хотя в стандартном коде этого может не быть).
Запуск: SoftGym рассчитан на работу в Python. Можно запустить в среде Kaggle, если доступен **GPU с CUDA**. Есть Docker-образ и инструкции [16] [17]. Для примера, чтобы увидеть случайную симуляцию, команда: `python examples/random_env.py --env_name ClothFoldCrumpled` запускает среду со **скомканной тканью** [18]. Появится окно с графикой (если доступен дисплей) или можно сохранить кадры. Blender не требуется – всё через PyFlex. Требуется CUDA 9.2/10 и драйвер NVIDIA [19]. В Kaggle (Ubuntu) можно попытаться установить, хотя настройка непростая. Тем не менее, SoftGym – открытый проект, так что запуск воспроизводим на локальной машине с GPU.
Визуальный результат: Да, на сайте проекта есть GIF-анимации (например, *SpreadCloth*: разжатие смятой ткани) – видно реалистично **помятый кусок материи** [20]. SoftGym генерирует глубокие складки, заломы и даже контакты слоев ткани, аналогичные тому, как выглядела бы **скомканная бумага или ткань**. Эти состояния используются для обучения, но сами по себе являются ценными примерами смятия. Таким образом, SoftGym предоставляет готовые mesh-состояния и динамику сминания/расправления.

Источник: SoftGym README (Cloth tasks, "crumpled cloth" environments) [12] [14]

6. OpenCloth – сборник алгоритмов симуляции ткани (OpenGL)

Проект: OpenCloth – открытый код, реализующий множество алгоритмов симуляции ткани и одежды. Хотя он не посвящён специально бумаге, некоторые режимы позволяют получить эффект смятой бумаги.
Репозиторий: mmmovania/opencloth [21] [22].
Технологии: C/C++ с использованием **OpenGL** для расчёта и визуализации. Поддерживается CPU и GPU (есть версии на GLSL, OpenCL, CUDA) [23].
Реализация скомкивания: OpenCloth предоставляет реализацию разных моделей – от простого

массо-пружинного полотна до продвинутых интеграторов и **позиционно-ориентированной динамики (PBD)** [22]. Для имитации бумаги наиболее релевантны модели с высокой жёсткостью на растяжение и, возможно, учётом пластических деформаций. В списке есть, например, **Implicit Euler с большими коэффициентами жесткости** или **Co-rotated FEM** – они могут поддерживать очень жёсткий материал. С помощью OpenCloth можно воспроизвести сценарий: плоскость (ткань) падает на сферу или сжимается между объектами – при высоком модуле упругости это даст **острые заломы**. В проекте также реализован **ветер и силы** – можно "сдуть" ткань в угол, она там сомнётся. OpenCloth предназначен для обучения и экспериментов: код открыт и простой, можно включить/выключить самоколлизию, порвать ткань и т.п. Отдельно отмечено, что есть **демо с наложением текстуры и освещением** для явного интегратора [22] – то есть можно видеть морщины визуально.

Текстурирование: Поддерживается. Один из режимов – *Explicit Euler integration with texture mapping* [22]. Это означает, что на полотно накладывается изображение (UV-маппинг), и при симуляции оно деформируется вместе с сеткой. Таким образом, **можно натянуть, например, картинку на лист** и наблюдать, как она сминается вместе с бумагой.

Запуск: Проект предоставляет исходники и даже готовые бинарники под Windows [24]. Для запуска на Kaggle потребуется скомпилировать (библиотеки OpenGL, GLUT). Возможно, в среде без графики придётся использовать эмуляцию OSMesa. В Windows можно просто запустить `.exe` из папки `bin` [24] – откроются демо. Код рассчитан на обучение, поэтому "из коробки" есть готовые сцены: достаточно двойным кликом открыть исполняемый файл демо (например, Demo1, Demo2) [25]. Эти демо показывают падающее полотнище, взаимодействие с шаром и т.п. Чтобы смять лист, можно изменить сцену – например, зафиксировать несколько краёв и столкнуть их. Благодаря открытости кода, несложно прописать сценарий сжимания.

Визуальный результат: Да. Есть видео-демо на YouTube [25]: ткань мнётся, колышется ветром. С высокой жёсткостью получится **комок с острыми складками**. OpenCloth ориентирован на наглядность: все алгоритмы просто реализованы в одном файле, можно экспериментировать и видеть разницу. Для задачи бумаги – ценен как *"кухня алгоритмов"*: можно попробовать классический масс-пружинный подход или PBD и сравнить, какие складки получаются. Итоговые анимации не привязаны к Blender, но их можно визуализировать с текстурами и светом (прямо в OpenGL) [21] [22].

Источник: OpenCloth README (описание целей и возможностей) [21] [22]

7. OpenClothPy (Taichi-библиотека для симуляции ткани)

Проект: OpenClothPy – попытка реализовать cloth-симуляцию средствами Python на основе фреймворка **Taichi** (язык для вычислительной графики), вдохновленная OpenCloth.

Репозиторий: lyd405121/OpenClothPy [26] [27].

Технологии: Python, **Taichi** (с бэкендом Vulkan или CUDA) – позволяет писать высокопроизводительный параллельный код для симуляции на GPU прямо на Python.

Реализация скомкивания: Код проекта довольно компактный (файл `cloth.py`), использует Taichi для вычисления сил и перемещений точек ткани. Вероятно, это масс-пружинная модель, работающая на GPU. Проект объявлен как прототип: нужно установить зависимости и **Vulkan SDK**, затем запустить командой `ti cloth.py` (ti – командный интерфейс Taichi) [26] [27]. При запуске мы получим анимацию полотна (в `cloth.gif` видно колыхание ткани, возможно, падающей на сферу) [28]. Чтобы воспроизвести смятие бумаги, можно модифицировать начальные условия – например, задать листу большую жесткость и применить несколько импульсных сил (как удары), чтобы он помялся. Taichi упрощает эксперименты – код на Python, но работает быстро за счет компиляции в шейдеры.

Текстурирование: В чистом OpenClothPy, вероятно, нет текстурирования (он рисует ткань одноцветной). Однако, поскольку можно экспортировать результирующие вершины или

визуализировать через taichi GUI, теоретически можно раскрасить вершины по UV или передать текстуру шейдеру. Простого способа “натянуть картинку” может не быть, но получить координаты вершин и использовать их для UV-маппинга вне Taichi – можно.
Запуск: Требуется Python среда с **Taichi** (подойдет Kaggle, если есть поддержка Vulkan – на Kaggle можно установить `taichi` через pip). После установки Vulkan SDK (для Windows) или использования встроенного Vulkan (Linux), запускается командой `ti cloth.py` [29]. Это откроет окно с симуляцией. На Kaggle без дисплея вывод можно перенаправить в GIF. Проект небольшой, так что его легко модифицировать: например, в `cloth.py` задать гравитацию, точки закрепления и добавить столкновение с невидимыми плоскостями, чтобы смять лист.
Визуальный результат: Частично. В репозитории есть `cloth.gif` – анимация ткани. Она демонстрирует **динамическую деформацию полотна**. При настройке параметров можно получить либо мягкое колыхание (как ткань), либо почти бумажное поведением (резкие перегибы). Так как Taichi поддерживает столкновения и ограничения, при некотором кодировании можно имитировать комканье: например, сжать ткань с четырёх сторон. Визуально результат будет представлен как набор точек/треугольников (рендер примитивный). Но главное – проект даёт кодовую базу, где можно экспериментировать с моделью ткани, чтобы добиться эффекта **скомканной бумаги**, используя возможности GPU.

Источник: OpenClothPy README (способ запуска с Vulkan/Taichi) [26] [27]

8. PyBullet – симуляция мягкого тела (cloth) с помощью Bullet Physics

Проект: Пример использования физического движка **Bullet** (через Python-API PyBullet) для симуляции листа как **мягкого тела** (soft body). PyBullet позволяет загружать меш как мягкое тело и рассчитывать его деформацию и столкновения в реальном времени.
Репозиторий/Ресурс: Код-образец на StackOverflow (вопрос про cloth в PyBullet) [30] [31]; также *PyBullet Quickstart Guide* описывает deformable объекты [32].
Технологии: Python (PyBullet), Bullet Physics 2. **(с поддержкой soft body и cloth).**
Реализация скомкивания: **В PyBullet можно вызвать** `p.loadSoftBody("cloth_z_up.obj", ...)` **для загрузки меша ткани (например, сетка 31x31 вершин) как мягкого тела [30]. В параметрах указываются коэффициенты: включение** массо-пружинной модели **(useMassSpring=1), упругость пружин (springElasticStiffness), демпфирование, включение изгибающих связей (useBendingSprings=1) и самоколлизии [33]. После загрузки мягкого тела можно создавать обычные твердые коллайдеры (стены, сферы). Далее запустить симуляцию шаг за шагом:** `p.stepSimulation()` **в цикле [31]. Чтобы** скомкать лист**, можно, к примеру, зафиксировать углы листа и сблизить их (imitating a person crumpling paper), или прижать лист между двумя сходящимися плоскостями. Bullet просчитает смятие – меш согнется, образуются складки, при хорошей настройке включится** face contact **(треугольники не проходят друг сквозь друга) [34]. В примере на SO пользователь накладывал два слоя ткани и замечал, что они перемешиваются – это указывает на сложности самоколлизии, но для одного листа будет проще. Суть: PyBullet предоставляет** полноценную физику мягкого тела **– аналог Blender cloth, но доступную программно.**
Текстурирование: **PyBullet сам по себе рендерит примитивно (RGB пиксели без сложных материалов). Однако, можно получить меш (через** `getMeshData` **) и затем отрисовать его с текстурой во внешнем движке (Open3D, Panda3D и пр). Либо использовать PyBullet GUI: там мягкое тело отображается, но без текстуры. В целом,** применение текстуры возможно, если после симуляции экспортировать деформированный меш и загрузить в движок, поддерживающий UV-маппинг **(например, Blender или Unity).**

Запуск: **PyBullet – кроссплатформенная библиотека, легко ставится через** `pip`**. На Kaggle (Python) её можно использовать. Нужен OBJ-файл сетки бумаги (например,** `cloth_z_up.obj` **из примеров Bullet). Минимальный код: инициализация PyBullet,** `p.resetSimulation(p.RESET_USE_DEFORMABLE_WORLD)`**, установка гравитации, загрузка плоскости-коллайдера и soft body, затем цикл симуляции** [35] [30] [31]**. Можно включить PyBullet GUI для визуализации или собирать данные и потом визуализировать. Для ускорения симуляции большого сгиба стоит уменьшить размер шага, повысить итерации солвера, возможно, включить** CLoth vs Cloth collision **(PyBullet пока ограничен, selfCollision=0 или 1).**
Визуальный результат: **Да. PyBullet демонстрировал примеры:** ткань, опадающая на сферу **(образует складки) и** захват полотна "руками" **(позволяет сложить его). При достаточной жесткости пружин (springElasticStiffness ~ 100) и малой растяжимости, мягкое тело Bullet будет** сминаться с резкими складками**, напоминая бумагу. Например, если тянуть противоположные края квадрата в разные стороны, а потом сблизить – получится комок. На выходе можно сохранить последовательность состояний (точек) – т.е. получить анимацию смятия. В целом, PyBullet – мощный инструмент для этой задачи, с открытым кодом и активным сообществом.

Источник: Пример кода PyBullet (loadSoftBody для ткани, цикл симуляции) [30] [31]

9. Argus – адаптивная симуляция тонких листов с трением (на базе ARCSim)

Проект: Argus – научный проект (SIGGRAPH 2018) по симуляции одежды с учетом контактного трения и адаптивного ремешинга. Включает код для **складывания и смятия листов** с высокой точностью (развитие идеи ARCSim).
Репозиторий: lijieumn/argus-distribution [36] [37] .
Технологии: C++ (с Python-интерфейсом для запуска примеров), используется модифицированный **ARCSim** – продвинутый решатель для тонких оболочек, + библиотека **So-Bogus** для сухого трения [38] .
Реализация скомкивания: Argus предназначен для физически корректной симуляции тканей и листов. Он использует **адаптивную сетку** (переразбиение треугольников в местах сильной кривизны) – это значит, при смятии появляются новые вершины, захватывающие мелкие складки. В Argus реализована поддержка **пластических деформаций** – т.е. лист может сохранять форму заломов. Базируется на работах Rahul Narain и др., в частности, прямо цитируется статья *"Folding and Crumpling Adaptive Sheets"* (SIGGRAPH 2013) [37] – в ней описывалась симуляция бумаги, включая образование характерных "вершин" складок. Таким образом, Argus способен **правдоподобно сминать бумагу**, соблюдая неизменяемость длины волокон (paper is inextensible) и формируя острые сгибы. В репозитории есть готовый пример: `box_and_cone.py` – квадратный платок между коробкой и конусом [39] . При запуске, квадрат адаптивно сгибается вокруг препятствий. Нажатием *Space* можно увидеть его постепенное смятие [40] . Также есть интерфейс `argus_interface.py` для сохранения кадров (OBJs) анимации и headless-режима [41] . Пользователь может настроить свойства материала (модуль Юнга, коэффициент пластичности). За счет учета трения складки не скользят и не раскрываются – бумага "держит" форму.
Текстурирование: Argus – симулятор, он не занимается визуализацией текстур. Но он сохраняет **последовательность OBJ-мешей** (или PNG кадры) [42] . Эти OBJs можно загрузить в Blender/Unity и наложить на них заранее подготовленную UV-развертку с текстурой (например, если известна развертка исходного плоского листа, её можно применить и к деформированному мешу). Так что, косвенно – да, текстуру натянуть возможно.

Запуск: Проект нужно скомпилировать (CMake + зависимости). Он рассчитан на Linux/Ubuntu, но можно собрать и на Windows с усилиями. Для запуска примеров используются Python-скрипты. Например: `python examples/box_and_cone.py` – откроется окно с квадратным листом над объектами [39]. Нажимаем пробел – начинается симуляция сминания. Или `python examples/argus_interface.py --headless --save_objs` для пакетного просчета и сохранения мешей [42]. В Kaggle этот проект тяжел для установки (нужен компилятор, Eigen, Cholmod и прочие зависимости, а также GPU для быстроты), но возможен при наличии времени. В комплекте есть несколько примеров "character cloth", но для бумаги достаточно валидATION пример. Argus – не интерактивный GUI-инструмент, а исследовательский код; тем не менее, он публичный и повторимый.
Визуальный результат: Да, и очень качественный. Судя по публикациям, Argus/ARCSim могут **точно воссоздать мятую бумагу** – с треугольными узорами складок, характерной жёсткостью. В Validation показано сравнение: симулированный лист и реальный – они очень схожи. Можно сохранить анимацию процесса сжатия. Например, в статье 2013 г. была GIF, где виртуальный лист **скомкивается руками в ком** – Argus способен повторить нечто подобное, учитывая трение (листы не проскальзывают друг по другу). Таким образом, Argus – наиболее физически реалистичный (но и ресурсоемкий) пример симуляции скомканной бумаги [37].

Источник: Argus README (адаптивная cloth simulation, ссылка на "Folding and Crumpling Adaptive Sheets") [38] [37]

10. Алгоритмическая генерация смятого листа (проект Hoffmann et al. 2019)

Проект: Код из исследования *"Machine learning in a data-limited regime: Augmenting experiments with synthetic data uncovers order in crumpled sheets"* (Hoffmann et al., Science Advances 2019) – предназначен для генерирования синтетических данных о смятой тонкой пленке.
Репозиторий: hoffmannjordan/Crumpling [43] [44].
Технологии: C++ (алгоритм симуляции плоского складывания), Python и Mathematica (анализ).
Реализация скомкивания: Проект фокусируется не на динамической анимации, а на **генерации геометрии смятого листа** с заданными статистиками. В частности, включён C++ утилита `flatfold_gen` [45]. Этот генератор принимает параметры: число складок, случайный seed, доля радиальных сгибов и пр. [45]. Он закладывает серии сгибов (折叠) в плоскость, имитируя процесс сжатия – но делает это *квазистатически*, вычисляя финальную сетку. Результат – набор данных (.dat) о вершинах и ребрах листа с указанными "горами" и "долинами" сгибов. Скрипт `run.py` позволяет запускать генерацию с разными параметрами и строить изображения результатов (через gnuplot) [45]. Хотя динамики нет, **полученная 3D форма соответствует смятому листу** (в работе сравнивают синтетические складки с отсканированными реальными и используют их для тренировки ИИ) [46]. Интересно, что проект не полагается на физический движок, а случайным образом генерирует разворот сгибов, используя правила оригами (возможно, опираясь на Voronoi-декомпозицию, упомянута библиотека voro++). Тем не менее, конечная геометрия довольно реалистична: сохраняется развёртываемость поверхности (нет растяжения). Эта программа фактически **эмулирует многократное сложение листа** (как если бы мы скомкали его руками, но без учета упругого возвращения).
Текстурирование: Код выводит либо данные для построения кривизны, либо координаты сетки. **Меша (.obj) напрямую не генерируется**, но можно модифицировать код, чтобы сохранить треугольную сетку смятого листа. Авторы концентрировались на статистике складок, а не визуализации, поэтому вопрос текстур не рассматривается. Однако, зная структуру (вершины/ребра, какая сторона "лицевая"), можно наложить условно текстуру бумаги (например, белый с шумом) для визуализации.

Запуск: Код рассчитан на **Ubuntu**. Нужно скомпилировать `sim_flatfold.cc` (Makefile прилагается). После этого запускается `./flatfold_gen <NUM_FOLDS> <SEED> <FRACTION_RADIAL> <ALL_SAME_DIR>` [45]. Например: `./flatfold_gen 50 42 50 0` – сделает 50 случайных сгибов, половина радиальных, случайные направления. Он создаст два .dat файла и скрипт gnuplot `fold.gnuplot` [45]. Запустив его, получим `fold_fig.png` – визуализацию складок. Для пакетного запуска есть `run.py`, который генерирует изображения для серий параметров [47]. Поскольку это научный код, придётся внимательно читать README. На Kaggle возможно запустить, если установить компилятор и gnuplot. Взаимодействие с Blender не требуется.

Визуальный результат: Да, но в специфической форме. Результаты – изображения высотной карты и кривизны смятого листа [46]. В статье показаны примеры: **сгенерированный плоскостной паттерн смятого листа** очень похож на реальный – сеть треугольных складок, пересекающихся под ~120°. То есть, проект Hoffmann генерирует не плавные морщины, а именно **острые заломы**, что соответствует бумаге. Анимации процесса нет (это статическая конечная форма). Тем не менее, имея такой алгоритм, можно породить много разных форм смятой бумаги для рендеринга или обучения нейросетей. Таким образом, это пример процедурной генерации *геометрии* скомканной бумаги, альтернативный физическим симуляциям.

Источник: Hoffmann et al. README (утилита flatfold_gen для генерации смятого листа) [45]

Вывод: Существует множество подходов к задаче 3D-анимации сжатия бумаги – от встроенных средств Blender (cloth и soft body симуляции) до специализированных физических движков (Bullet, FleX, ArcSim) и даже процедурных генераторов. Выбор зависит от требуемого баланса между физической реалистичностью и простотой. Все перечисленные проекты предоставляют исходный код или файлы, позволяющие воспроизвести эффект; большинство позволяют наложить текстуру (через UV-карты) и сохранить либо визуализировать финальную анимацию. Визуальные примеры (рендеры или видео) имеются практически для каждого – будь то кадры обучающих видео Blender [1], GIF-анимации из научных проектов или демо-видео GitHub-репозиториев – и они демонстрируют, что скомканная бумага может быть достоверно смоделирована средствами компьютерной графики и физики.

---

[1] How to Crumple Paper in Blender With a Cloth Sim - Lesterbanks
https://lesterbanks.com/2019/09/how-to-crumple-paper-in-blender-with-a-cloth-sim/

[2] Blender Destruction Tutorial: 9 Ways to Destroy Things • Creative Shrimp
https://www.creativeshrimp.com/blender-tutorial-9-ways-to-destroy-things.html

[3] modeling - paper simulation - Blender Stack Exchange
https://blender.stackexchange.com/questions/233175/paper-simulation

[4] [5] Modelling crumpled paper - Modeling - Blender Artists Community
https://blenderartists.org/t/modelling-crumpled-paper/655441

[6] [7] [8] [9] [10] GitHub - BinRoot/Blender-Cloth-Simulation: Simulates a cloth being held from random points (for robot cloth folding)
https://github.com/BinRoot/Blender-Cloth-Simulation

[11] [12] [13] [14] [16] [17] [18] [19] GitHub - Xingyu-Lin/softgym: SoftGym is a set of benchmark environments for deformable object manipulation.
https://github.com/Xingyu-Lin/softgym

15 20 SoftGym
https://sites.google.com/view/softgym/home

21 22 23 24 25 GitHub - mmmovania/opencloth: A collection of source codes implementing cloth simulation algorithms in OpenGL
https://github.com/mmmovania/opencloth

26 27 28 29 GitHub - lyd405121/OpenClothPy
https://github.com/lyd405121/OpenClothPy

30 31 33 34 35 python - cloth-like objects in Pybullet - Stack Overflow
https://stackoverflow.com/questions/78600422/cloth-like-objects-in-pybullet

32 [PDF] ROS-PyBullet Interface - Proceedings of Machine Learning Research
https://proceedings.mlr.press/v205/mower23a/mower23a-supp.pdf

36 37 38 39 40 41 42 GitHub - lijieumn/argus-distribution
https://github.com/lijieumn/argus-distribution

43 44 45 46 47 GitHub - hoffmannjordan/Crumpling: Code from "Machine Learning in a data-limited regime: Augmenting experiments with synthetic data uncovers order in crumpled sheets" Science Advances 2019
https://github.com/hoffmannjordan/Crumpling