In-Place 3D Sketching for Authoring and Augmenting Mechanical Systems

Oriel Bergig*, Nate Hagbi*, Jihad El-Sana*, Mark Billinghurst†

* The Visual Media Lab, Ben-Gurion University, Israel

†The HIT Lab NZ, University of Canterbury, New Zealand

ABSTRACT

We present a framework for authoring three-dimensional virtual scenes for Augmented Reality (AR) which is based on hand sketching. Sketches consisting of multiple components are used to construct a 3D virtual scene augmented on top of the real drawing. Model structure and properties can be modified by editing the sketch itself and printed content can be combined with hand sketches to form a single scene. Authoring by sketching opens up new forms of interaction that have not been previously explored in Augmented Reality.

To demonstrate the technology, we implemented an application that constructs 3D AR scenes of mechanical systems from freehand sketches, and animates the scenes using a physics engine. We provide examples of scenes composed from trihedral solid models, forces, and springs. Finally, we describe how sketch interaction can be used to author complicated physics experiments in a natural way.

KEYWORDS: In-Place Augmented Reality, free hand sketching, Augmented Reality, 3D content authoring, physical simulation, interaction by sketching, visual language, dual perception.

INDEX TERMS: H.5.1 [Multimedia Information Systems]: Artificial, augmented, and virtual realities; I.4.0 [Image Processing and Computer Vision]: Image processing software; K.3.0 [Computers and Education]: Computer Uses in Education.

1 INTRODUCTION

Freehand sketching is one of the most ancient of human skills. It has been used from our early days as a natural communication language. Sketching facilitates conveying visual information, while encouraging creativity. It also serves as a natural way for triggering visual thinking, which is essential in many domains.

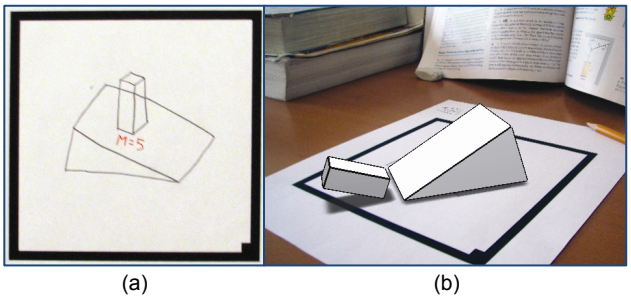
Various applications have exploited the power of computerized sketching, beginning with the pioneering work of Ivan Sutherland's, SketchPad [1]. Computerized sketching offers various advantages over freehand sketching and allows authoring of complex objects.

The visual language of physics is well defined and physics textbooks commonly include abstract diagrams of physical systems to explain the studied material. Teachers usually sketch physical systems on the class whiteboard, and students often sketch to solve physics problems. On the other hand, computer graphics and Augmented Reality (AR) enable three dimensional visualization of, and interaction with, physical systems as if they were real systems in a lab. In this work, we explore the combination of freehand sketching input and AR visualization for authoring physical systems. This combination can assist learning in ways that have not been explored before.

Authoring of three-dimensional scenes often requires extensive work. Several interaction techniques have been suggested for authoring physical scenes. Nevertheless, this remains a complicated task for the untrained user. On the other hand, sketching physical systems is easier to most people. State of the art methods exist nowadays for interpreting hand sketches and reconstructing three-dimensional geometric structures from line drawings. It is possible to exploit this knowledge to make the authoring of physical scenes easier and faster by sketching them in two-dimensions.

Interpretation of two-dimensional content for authoring three-dimensional scenes has been recently proposed in Augmented Reality. In-Place Augmented Reality (IPAR) [2] content is extracted from a printed-paper using a visual language. One of the key elements of IPAR is the dual perception property, which implies the visual language is understandable to humans without the use of any computerized system, but it also encodes AR content that can be extracted using computer vision methods. Orthographic two-dimensional projections of solid models are inherently a dual perception representation. They encode information about the 3D geometry of a model, which can be extracted and used as AR content. This work combines the advantages of IPAR content authoring with the advantages of sketching.

Automatic interpretation of hand sketches suggests a new form of interaction with content, which we name *sketch interaction*. With sketch interaction, three-dimensional scenes can be authored gradually by adding the 3D representation of sketch elements into the simulation. It is also possible to modify model geometry and properties by sketching. Manipulating models in a scene can be done by combining traditional interaction methods and sketching. For example, positioning models with fiducials and changing their geometry with eraser and pencil. The augmentation of the scene on top of its sketch makes sketch interaction intuitive, since users can observe scene modifications in the same place they are made.



(a) (b)

Figure 1. (a) Authoring a mechanical system by hand sketching on paper. The sketch is acquired by a webcam. (b) A virtual 3D scene is constructed, augmented and simulated on top of the sketch. The figure was taken during physical simulation.

* {bergig, natios, el-sana}@cs.bgu.ac.il

† mark.billinghurst@hitlabnz.org

IEEE International Symposium on Mixed and Augmented Reality 2009 Science and Technology Proceedings
19 -22 October, Orlando, Florida, USA
978-1-4244-5419-8/09/$25.00 ©2009 IEEE

In this paper, we present software that can analyze, visualize, and simulate mechanical systems in 3D (see Figure 1). We describe in chapter 3 our sketch interpretation and our scene composition algorithms. We would like to maintain the inherent advantages of the inaccuracy of the sketching process, while supporting reliable simulation using a simple hardware setup. Our main contributions include: (a) The ability to manually sketch a 3D mechanical system using a regular pencil and paper, and (b) Sketch interaction - a new conceptual approach for interaction with virtual content.

Analyzing 2D sketches of physical scenarios and generating a corresponding 3D physical scenarios is a challenging task. The current implementation is limited to simple orthographic projections of trihedral solid models with a limited set of physical properties.

The next chapter describes related work. Chapter 3 discusses the application use and elaborates about challenges faced, as well as implementation details. Chapter 4 describes the system setup and our experiments. Chapter 5 discusses the usability of sketch-based AR systems for understanding mechanics concepts, and Chapter 6 concludes.

2 BACKGROUND

This work builds upon a variety of concepts and algorithms developed in several domains. We first describe related work in Augmented Reality. We then turn to describe related work in sketch interpretation, focusing mainly on methods developed for educational applications. Finally, we review related methods for geometric reconstruction of 3D models from 2D sketches.

In a recent paper [3], Eric Klopfer *et al.* claim "most educators will agree that immersing students in a learning experience that allows them to grapple with a problem, gaining higher-order thinking skills from pursuing the solution, is a key aspect in good learning". Augmented Reality is clearly an effective way for immersing students in learning. Several Augmented Reality applications have been developed in the past to support learning. PhysicsPlayground [4] is an Augmented Reality application for mechanics education. Students can actively build their own experiments using a wireless pen and the personal interaction panel (PIP), and study them in a three-dimensional virtual world using the variety of tools and features provided. Using PIP a student can insert basic shapes into the scene, scale, rotate, and translate them. A similar hardware setup and authoring technique has been used in Construct3D [5], a three-dimensional geometric construction tool, specifically designed for mathematics and geometry education. Teachers and students were highly motivated to use the application. However, as stated by the authors, practical utilization in schools was hindered by hardware costs, support of a low number of simultaneous users, and the technical complexity of the setup [4]. In our work, we combine advantages inherent to immersive environments with simple hardware setup.

Other researchers have also demonstrated the ability to author AR models and interact with them while augmentation takes place. ARpm, is an Augmented Reality interface for polygonal modeling using 3D Studio Max [6]. BuildAR [7] provides a graphical user interface that simplifies the process of authoring AR scenes. The first step is loading existing 3D model files onto AR markers. The user can then apply geometric transformations to them. Henrysson et al. [8] describe and evaluate AR manipulation techniques for mobile phones and present a pilot user study. In our work, we allow the manipulation of models in a similar fashion, but the authoring of models and scenes is done by hand sketching on paper or a white board.

Sketch understanding is a fertile research field on its own. The Sketch Understanding Group at MIT is working toward a kind of "magic paper" [9]. They first analyze sketch primitives and then recognize the shapes according to geometric constraints, using the sequence of strokes and shape appearance. One of this group's early systems, ASSIST [10], performs interpretation of informal sketches of 2D physical systems and simulates them in real time. ASSIST represents hand drawn sketches as a sequence of strokes, which are analyzed while the user sketches them. This introduces time-dependant information, assisting the sketch interpretation process. For example, corners are located by analyzing stroke curvature and drawing speed, as people commonly slow down when drawing corners.

The Smart Tools Lab at CMU presented an optimized sketch recognition engine for easy training [11]. They developed a symbol recognizer that uses an image-based recognition approach, allowing learning a symbol definition from a single prototype example [12]. In another work [13] they demonstrated the use of constraint satisfaction techniques for efficiently constructing physically consistent interpretations. They demonstrated that for schematic sketches of physical systems, even simple physical reasoning can overcome the inaccuracies and ambiguities found in freehand sketches.

Farrugia et al. [14] demonstrated a sketch-based interface that combines the portability of pen and paper with that of camera-phones. A user sketches a planar projection of a model using a prescribed sketching language. The image is then captured and sent for processing on a remote server, which sends back a 3D representation of the image. The authors conducted a user study indicating the need for a mobile computational tool to obtain visual representations of 3D geometric models from freehand sketches. In our work, in a fashion similar to [14], we address sketching on paper with ordinary pencil. However, we support interactive authoring of physical scenarios in real time. Since the AR scene we construct is fully three-dimensional, the existing reasoning methods, e.g., [9-13], must be adjusted to fit our sketch interpretation scheme.

The human vision system can interpret a single two-dimensional line drawing as a 3D object without much difficulty, even when some of its lines are invisible due to occlusion. This problem, named geometrical reconstruction, requires determining the geometric and topologic relations of all the atomic parts of an object, which can be represented by a connected graph. Naturally, a single line drawing may have several plausible interpretations. A distinction is made between methods that interpret drawings without hidden lines and methods that deal with natural line drawings. A detailed survey by Company et al. [15] categorizes existing methods, and a recent paper by Liangliang et al. [16] cites the most updated ones.

Existing methods use as input a vector-based representation of line drawings. Lipson et al. [17, 18] address the interpretation of wireframe models, whose occluded lines are visible in the sketch. We found their algorithm suitable for the reconstruction of model structure for sketched physical system elements. However, translation of a sketch image into a vector-based representation introduces several difficulties in cases of complex wireframe models. For example, sketches containing a large number of hand drawn strokes in close proximity are often difficult to interpret when the camera resolution is relatively low. On the other hand, the methods proposed by Varley et al. [19-22] deal with natural line drawings, where occluded lines are culled. For mechanical scenarios, where assumptions about model geometry can be made, these methods are highly applicable. We assume that stroke junctions are of degree two or three, and that neighboring faces are perpendicular if the angle between their normals is nearly right.

In this work, we develop methods to interpret hand-sketches and construct the corresponding 3D scenes. We require a simple hardware setup, where sketches are captured with a simple camera

without the need to track user strokes while they are being drawn. We construct a mechanical scene from several objects, and position them in 3D even in cases they overlap in the sketch. Physical properties are inferred from annotations provided near the objects and physical simulation is performed for the 3D models, represented as meshes. We augment the scene on its sketch and suggest a new interaction technique. Manipulation of models is performed on the virtual content, while modification to model structure and properties are applied to the sketch itself.

3 In-Place 3D Sketching of Mechanical Systems

In-Place 3D Sketching of Mechanical Systems is an application that captures sketches of mechanical systems, analyzes them to compose a 3D scene representing the system, and finally augments and animates the scene using a physical simulation engine. The sketches may already exist in a printed book or they can be sketched by hand. The user can interact with the application by sketching additional elements, modifying existing sketch elements, or manipulating 3D models using the camera. We first describe the use of the application and then the application pipeline.

3.1 Application

The application starts by capturing video from a webcam and displaying it to the user. We use a black frame for registration and pose estimation. Once the black frame is located in the video, the layout inside the black frame is analyzed, and a sketch of a mechanical system is searched.

3.1.1 Authoring a 3D Physical System

The user sketches a mechanical system that contains orthographic projections of physical elements (such as inclined planes) and can use annotations (such as arrows, springs, and equations). In the current implementation, we support trihedral solid models with occluded lines culled, drawn in black, and a limited set of physical and geometric annotations drawn in red.

Given a sketch image, the application constructs a three-dimensional representation for it. The user may sketch the scene step by step, adding models and physical properties to an already analyzed and augmented system. The user can also see the result of each step augmented and animated before moving to the next step. Models can be set to be static, meaning they are not moving during physical simulation, or dynamic. The application handles physical elements that interact with each other in the sketch, e.g., an inclined plane and a box located on top of it. Elements that overlap in the sketch must be either sketched in different colors or introduced to the application in different steps. The user can manipulate models using traditional interaction methods, such as the camera and a single button, based on the method proposed in [8]. The transformation type has to be selected, and then camera movement is used for determining the transformation parameters.

3.1.2 Sketch Interaction

Sketching content for augmentation opens up new forms of interaction. The user can refine a scene by translating, rotating, and scaling models in the AR view, as well as by modifying their sketch on the paper. The modification performed updates the model structure. The user can modify physical annotations, which affect the physical properties of the model. For example, one can start with a simple block sketched on an inclined plane. A 3D block is augmented and simulated in 3D. The block can be repositioned, rotated or scaled to test different scenarios. The user may then wish to add friction to the inclined plane under the block and the simulation will change accordingly (see Figure 1 and 2). The user may then wish to change the block into a box as depicted in Figure 2(c). This is done using an eraser to delete lines, and a pencil for adding the required strokes. The augmented block model then turns into a box. This capability can be used to ease the authoring of the scene since the user does not have to reposition a model in the scene after modifying its structure.

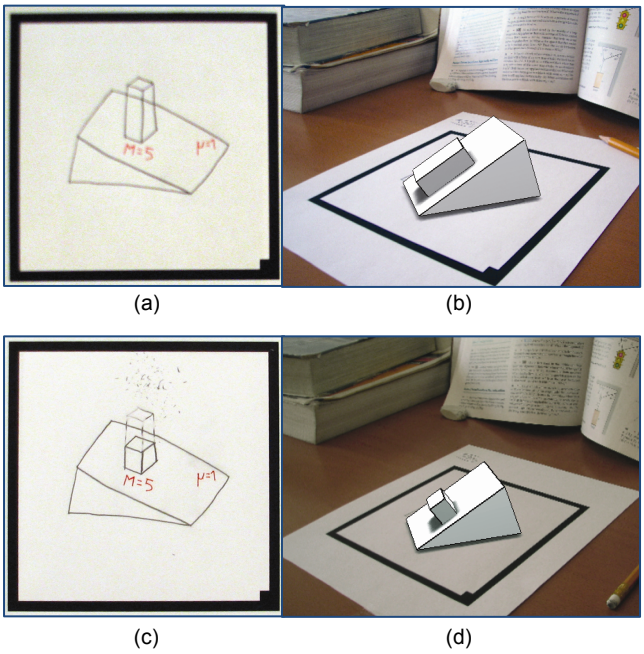


Figure 2. Sketch Interaction. (a) Friction is added to the inclined plane from figure 1. (b) The simulation is affected. (c) The block in the sketch is modified with eraser and pencil and turns into a box. (d) The block model turns into a box.

Content may be combined from different sources as well. The user may start by constructing a scene from a figure in a textbook and watch it augmented and simulated. The user may then wish to experiment with additional elements. To achieve this, he can sketch new elements on paper and combine them transparently into the textbook scene.

3.2 Application Pipeline

The application pipeline starts with capturing a live video from a webcam and producing an augmented and animated three-dimensional scene. The application consists of four main components, *Image Understanding*, *Structure Reconstruction*, *Scene Composition* and the application's *Main Loop* as depicted in Figure 3. The *Image Understanding* component enhances the input image and transforms a captured image of a mechanical system into its vector-based representation, with additional information used for further processing. The *Structure Reconstruction* component constructs 3D models from the generated vector-based representations. The *Scene Composition* component generates a scene graph and a corresponding virtual scene. Finally, the application's *Main Loop* performs the augmentation of the virtual scene, animating it according to its physical properties.

3.2.1 Image Understanding

The Image Understanding component enhances the image of a mechanical system and extracts objects and annotations. Objects are defined as elements of the mechanical system and annotations are defined as sets of symbols used in physics to express object properties. Finally, the strokes of each object are analyzed to yield a connected graph. We next describe each of these steps in detail.

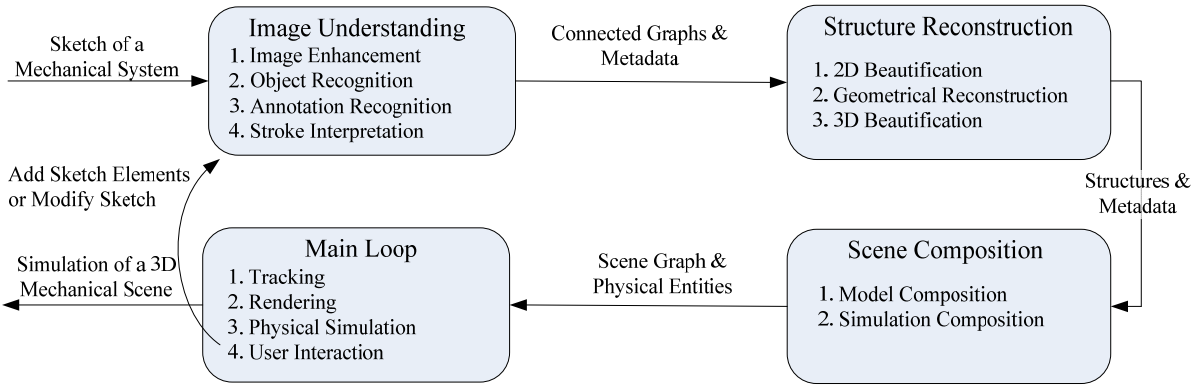


Figure 3 – The pipeline of In-Place 3D Sketching of Mechanical Systems

Image Enhancement

The aim of this step is to obtain an image from which strokes can be robustly extracted. The input sketch can be captured using a standard webcam supplying a live video feed, and hence the captured frames quality varies widely. In addition, the image can be captured from different angles, which implies the sketch may be projectively warped. First, we unwarp captured frames by multiplying them by the transformation calculated from the black frame. We then sample and average several registered frames and normalize the grayscale result to its entire intensity range, as proposed in [2]. We assume the sketch includes distinctive lines along the wireframe of each object. To emphasis the wireframes, we employ adaptive thresholding.

Object and Annotation Recognition

We start by identifying disjoint wireframes by analyzing the connected components in the image. To be able to robustly separate between overlapping objects, we require them to either be sketched in different colors or be added to the scene one by one. In the latter case, image difference and morphological operations allow separating between the objects. We subtract the image captured in the previous step P from the image of the current step C and filter disconnected pixels. The object introduced in C remains clear but pixels are missing where overlapping took place. To complete these missing parts, we dilate the new object using P as a mask.

Objects are usually constrained by annotations, such as letters and arrows. We perform rotation invariant recognition using shape contexts [23] against a small set of predefined annotations, and assign each identified annotation to its nearest object. We support annotations for mass, friction, force, spring constant, gravitation, and geometric properties for marking hollow objects and symmetric objects. If no object is found near an annotation, and if possible, the annotation is assigned to the entire scene.

Annotations may be related to other annotations or with more than one object. For example, an arrow annotation is assigned to the nearest object that intersects the arrow line, but also to the annotation of the force written on top of it. A spring annotation is assigned to the spring constant and the two objects it connects.

Stroke Interpretation

Up to this point, each object has been represented as a set of pixels. We now turn to generating a vector-based representation for each object in order to allow its reconstruction in the next step. We construct a connected graph representing the wireframe image. A node in the graph corresponds to a stroke junction in the wireframe, and has a 2D position in the paper's 2D coordinate system, defined by the registration method. An edge in the graph corresponds to a stroke in the wireframe. We perform edge linking in order to fill small gaps and filter noise. We then perform a four-step algorithm: (a) identifying strokes and junctions, (b) constructing an initial graph and filtering artifacts, (c) refining the graph by splitting and joining edges and (d) assigning 2D coordinates to each vertex.

We now turn to describe the steps of the interpretation as depicted in Figure 4. Identifying which parts of the sketch are strokes and which are junctions can be done in different ways. We have found that thinning, as described in [24], is highly useful for simplifying this task. After thinning is complete, pixels that have two neighbors are marked as stroke pixels, and the rest of the pixels are marked as junction pixels. We then construct an initial graph by tracing strokes to locate their ends. Edges that do not end close to a junction are usually short and can be safely removed. In the next step, the graph is refined with junctions that were not identified. We perform incremental line fitting on stroke pixels, traversing from one of the edge ends until we reach a junction pixel.

When the fitting error exceeds a predefined threshold, we return to the first pixel where the error started to monotonically increase and split the edge at that pixel. Line fitting also assists in determining whether a junction of degree three is a T-junction, which is essential for correct structure reconstruction. Finally, we calculate a line equation for each edge and the intersection points between the edges. The center point of all the intersection points gives the coordinates of the vertex.

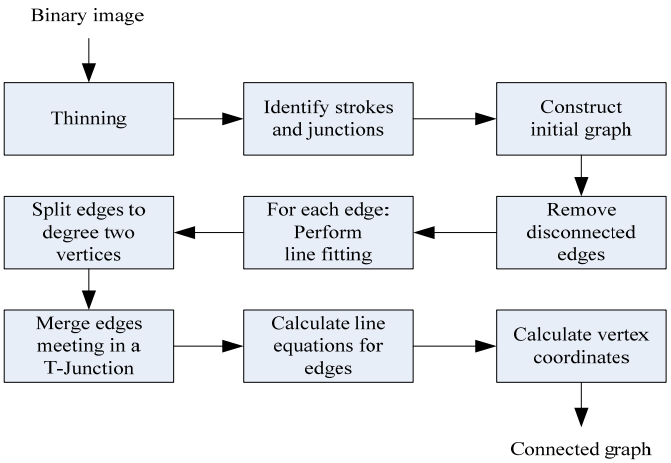


Figure 4: Stroke interpretation steps

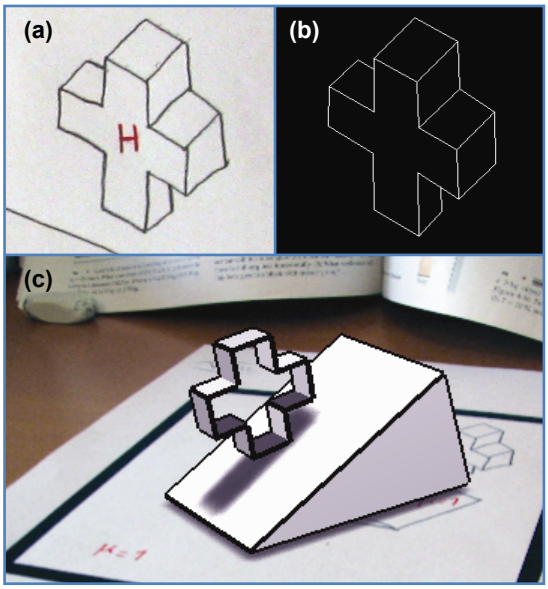


Figure 5: From sketch to 3D. (a) A sketched plus sign with an 'H' annotation signifying the object is hollow. (b) The beautified stroke interpretation of the model. (c) The 3D structure after structure reconstruction and 3D beatification on top of an inclined plane.

3.2.2 Structure Reconstruction

Structure Reconstruction is the process of inferring the three dimensional structure of a model from its connected graph representation. Beautification of the structure can be performed both on the two-dimensional connected graph before reconstruction, and on the 3D structure after reconstruction.

Beautification in 2D

Beautification overcomes sketch misalignments inherent to hand sketching, as depicted in figure 5(a). One prominent example is snapping straight lines with small angular deviations. In isometric sketches, most of the lines can be clustered by their direction. We identify the clusters using an angular histogram and decrease the deviation inside each cluster, which makes the lines parallel. Figure 5(b) depicts the results of performing stroke interpretation for figure 5(a) and then 2D beatification.

Geometrical Reconstruction

Next, we infer the 3D geometry of the sketched objects. Geometrical reconstruction from a sketch is a challenging task for which advanced algorithms have been proposed in the literature. We utilize the methods proposed by Varley [19]. First, we create the *frontal geometry*, where edges and vertices in the graph are assigned 3D positions according to regularities, and then construct the hidden regions. Each vertex is originally defined by its 2D coordinates, and the task of generating the frontal geometry amounts to determining the value of the third coordinate. The three most important steps in creating the frontal geometry are: (a) Line labeling - determining which lines in the drawing correspond to convex, concave, and occluding edges in the frontal geometry. (b) Line clustering - identifying sets of lines corresponding to parallel edge sets in 3D, and (c) Structure inflation - assigning depth according to compliance functions. We assume junctions are of degree two or three, which helps in generating the hidden regions. Generating hidden regions is based on projecting hypothesized edges and assigning confidence measures according to the certainty that edges intersect at the hypothesized location. The most plausible edge is then used, and the process is repeated iteratively. The interested reader is referred to [19] for more details.

Beautification in 3D

Another beautification step takes place after the geometry has been reconstructed. We look for triplets of faces sharing the same vertex. We calculate their intersection point and get a new vertex position that deviates from the original vertex position due to reconstruction errors. Another beautification technique we apply is snapping vertex coordinates to a 3D grid. Figure 5(c) depicts the geometrical reconstruction of the sketch in Figure 5(b) followed by the 3D beautification step.

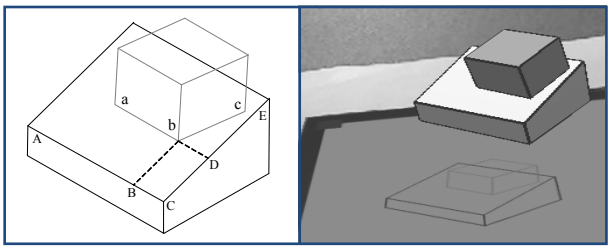
3.2.3 Scene Composition

In the Scene Composition step, we create a virtual scene graph where object structures are assembled into models with physical properties.

Model Composition

Our model representation includes its corresponding mesh, orientation, position, and additional properties. Properties are assigned using the keyboard or by sketching annotations. For example, an object may be hollow or solid. The orientation of a model with respect to its sketch can be calculated as the linear transformation between the base vertices of the 3D model to their corresponding vertices on the sketch. In practice, since the sketch is drawn in an isometric view and the 3D structure is rendered projectively after beautification steps, the sketch and the model are not aligned. A transformation may be calculated by minimizing the least square distance between the corresponding vertices. Instead, we split the positioning process of the model to steps. First, the base face of the structure is made parallel to the sketch plane. The base face is located by identifying the lowest strokes in the 2D orthographic representation. The translation of an object is set such that its center of mass projected on the image plane coincides with the center of the 2D orthographic representation of the model. Model orientation is defined according to an angle parameter since in many cases drawing a rotated model looks better.

Humans often draw orthographic projections of objects one inside the other or overlapping to indicate correlation. Since there is more than one possible assignment for the position and orientation of an object relative to another object, we make the basic assumptions described next. Figure 6 depicts two overlapping objects. The edge *abc* forms a face inside the face *ACE*. We assume the faces have the same plane equation and we measure the translation and rotation among these faces. Note that the position should be calculated in 3D coordinates, while the objects are drawn in 2D. Hence, the translation is measured in terms of distance ratios between the projections of vertices of one object on the edges of the other object. In Figure 6 the position of *b* relative to *C* is the ratio *BC/AC* in the direction $C \rightarrow A$ and *CD/CE* in the direction $C \rightarrow E$. The 3D position of b can then be calculated using edge ratios. *C* and b were chosen for determining the position of the smaller object since in the 3D structure, the angle *ACE* is closest to perpendicular and *b* is the closest to *C* in the plane. In a similar fashion, we calculate the relative rotation between the objects. In Figure 6 we measure the angle between the 2D line *bc* and the 2D line *CE*, and apply a 3D transformation using the vector normal to the faces as the rotation axis. These lines were chosen because they participate in the base *abc* and the other edges, *ab* and A*C,* are parallel. If they were not parallel, it would usually be perceived as if the box is rotated and the face formed by *abc* should not share the same plane equation with the face formed by *ACE*.



(a) (b)

Figure 6. Overlapping objects. Each object is geometrically reconstructed to a separate model. (a) Orthographic projection of two overlapping objects. (b) The composition of the models augmented on top of their sketch.

3.2.4 Simulation Composition

We attach to each model in the scene graph a physical entity that is composed of geometry, body, and properties. Models are represented as trihedral meshes and are set as static or dynamic by the user.

Annotations are used to set model properties, such as mass and friction, as well as properties of the entire virtual world, such as gravitation. We use default values in case essential simulation properties are not supplied. For example, surface friction defaults to zero, but one can write "$\mu=x$" to set the friction to $0.x$. We divide x by ten to make the friction annotation compact.

Arrows in the sketch and the force value sketched above them are treated as follows. We look for the face of the model the force should be applied to. We then assume it applies perpendicularly to that face through its center or it applies in 45 degrees from the middle of an edge. We continue the arrow line in both directions and find its first intersection with a sketch stroke, denoted s. We now need to choose between the two faces sharing s in the 3D model. If the arrow segment itself crosses another stroke then it is selected. Otherwise, we use the normals of the two faces. We project the normals on the 2D sketch plane and measure the angles between the projected normals and the arrow. If one of the angles is below a threshold, we assume the arrow applies to that face. Otherwise, we assume the force applies in 45 degrees on the middle of s.

Springs are treated in a similar fashion to arrows, operating on couples of objects. We first locate the two faces the spring connects. We then calculate the spring axis as perpendicular to one of the faces, termed the *leading face*. For example, in Figure 7 the leading face is the occluded face of the box. The connection point on the other face is calculated by intersecting the spring axis with that face.

3.2.5 Main Loop

The main loop of the application performs 3D pose tracking and renders the scene graph. It also runs the physical simulation engine and reacts to user interaction.

Physical simulation is performed using ODE [25], which is a physics engine designed for real time gaming. Simulation involves updating the orientation and position of each physical body introduced to ODE in the Simulation Composition step. In addition, arrows and springs cause forces to be applied to bodies in each simulation step. When two models are close to each other, joints are created between them. Physical properties, such as friction and bounciness, are then assigned to calculate the result of their interaction. After each simulation step, the physical entities update their corresponding models with their new rotation and position.

Simulation can be started, paused and stopped using the keyboard or using annotations. The user can sketch a triangle and a square, representing the 'play simulation' and 'stop simulation' commands. Introducing the triangle annotation into the scene causes the simulation to start. Introducing the square annotation stops the simulation.

4 Experiments

4.1 System Setup

One important advantage of the suggested application is its simple system setup as depicted in Figure 7. The application runs on a desktop computer connected to a standard webcam running in video mode. Sketching is performed using regular pencil and paper and the pencil does not have to be traced while sketching. Physical annotations are drawn using a red pen. The sketching area is surrounded by a black square used for registration and pose estimation, allowing 3D content to be augmented on top of the sketch. Figures from physics textbooks can be augmented as well, and we experimented with figures from [26]. We add wireframe lines to elements in the original book figures to simplify the recognition step.

Figure 7. System Setup. An inclined plane, a box, and a spring are augmented on top of their sketch. The setup consists of paper, a pencil, a red pen, and a computer connected to a webcam.

The hardware setup used in our experiments consisted of a laptop with a 1.83GHz Intel dual core processor, 1GB of RAM and a 128MB ATI graphics card. The system was running the Windows XP operating system. We performed experiments with two different cameras: a CMOS webcam that gives 960x720 resolution images and a CCD webcam that gives 640x480 resolution images.

The application was implemented in C++ using the Intel OpenCV [27] image-processing library, OpenGL for rendering, ODE [25] for physical simulation, and a modified version of the ARToolKit [28] library for registration and pose estimation.

4.2 Performance

We were mostly interested in finding system failure rates of interpreting and reconstructing models sketched by users. We asked five students to sketch cubes, boxes, and inclined planes until reconstruction succeeded without training them in advance. We found that sketching parallel edges as non-parallel is the main cause of failure. Reconstruction rarely fails for a box or a cube since these are usually sketched well. Reconstruction may fail for an inclined plane if the diagonal lines are not close enough to parallel. Inclined planes had to be sketched 1-3 times until they were properly reconstructed. As the users became familiar with the system sketching rules (parallel edges, no hidden lines) the failure rates decreased. Composing a complicated scene given as a

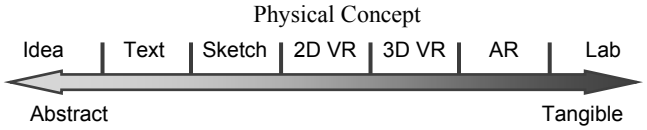


Figure 8. Physical Concept Continuum.

single sketch, such as the spring scene in Figure 7, yielded higher failure rates and sketch elements had to be redone 2 to 5 times before achieving the desired result. The main failure occurred when the system failed to extract the box wireframe correctly or when the box was poorly positioned.

We performed several failure tests independent of the users sketching skills. We experimented with different lighting conditions and reconstruction from different angles for a set of ten different sketches that vary in complexity. As long as the registration was not lost, the pose of the camera and the lighting had almost no influence on the reconstructed result.

We profiled the time it took for the system to run. The process of interpreting the sketch is performed once each time a new sketch is introduced and takes 1.3 seconds in average for simple models, such as cubes and inclined planes. It can take up to 2.1 seconds for complicated models (such as the one in figure 5). More than half of the time is spent on 3D reconstruction and more than a quarter of the time is spent on the sketch interpretation, depending on the number of edges present. The scene composition takes a few milliseconds. While not reconstructing a model the system operates at 25fps.

5 Usability

Some people find it difficult to understand certain physical concepts in mechanics. In many cases, when these concepts are explained in a more tangible way, they are easier to understand. Physical concepts can appear in different levels of abstraction (see Figure 8). The most abstract incarnation of a physical concept may be imagining a physical scenario. The next level of abstraction is formulating it in text. To make the text more tangible, we often sketch it. Without computer assistance, the next step is building an experiment in a lab, which might be demanding. Projecting Milgram's continuum [29] on the physical concept continuum, virtual reality is an abstract way of depicting a physical concept with computer assistance. The VR experience might contain a 2D GUI or even more realistically a 3D environment. Next on the continuum comes AR, followed by the lab experiment. In AR, the experiment appears in the view of the real world, which makes it even more understandable. Moreover, AR enables tangible interaction with the experiment.

Understanding physical concepts in each of the points on the continuum above is more effective when the user takes an active role than when taking a passive one. In many cases, to take an active part, students copy a sketch of a physical problem given in their textbook, even if they do not intend to change it. Building an experiment actively in a lab yields better understanding than watching such an experiment. Likewise, a computer program that gives its user the ability to create an experiment and play with it explains physical concepts more effectively.

Combining sketching and Augmented Reality makes the physical concepts even more tangible than when using AR by itself. In addition, this combination makes the user extremely active by allowing him to create the experiment by himself. Our experiments with the proposed system show that creating mechanical experiments using sketch interaction is a unique experience and a very tangible way for realizing mechanical concepts.

The system we implemented is limited in use. For example, reconstruction may fail for complex objects. Scene composition makes assumptions on the interaction between objects. In addition, the system does not perform detailed analysis of experiments, as done in [4].

The advantages of the simple system setup we propose with the ability to sketch naturally, are reflected best in simple and intuitive questions. We now turn to describe a use case of learning about friction using the implemented system we propose.

Use Case: Friction

High school physics books explain the fact that friction depends only on the friction coefficient and body weight, and does not depend on the contact surface area of the bodies. This fact is somewhat surprising. It means, for example, that the force that is required to cause a body to move over a surface remains the same, regardless of the face of the body in contact with the surface. We hence performed the following experiment. We sketched an elongated box, set its mass by sketching M=4 and applied a force by sketching an arrow pointing to it with an annotation of F=2. The friction coefficient was set to 0.45 and the gravity to 1.0. We ran the simulation a few times, flipping the box on its different faces. The box always moved in the direction of the force regardless the face area in contact with the paper surface. The box ceased to move, independently of its rotation, when we increased the mass by sketching M=5. This use case demonstrates how easily one can learn about a friction property using AR sketching.

6 Conclusion and Future Work

We introduced the ability to author mechanical systems in 3D by hand sketching using normal pencil and paper. We demonstrated the authored scene can be augmented and animated using a physical simulation engine. The mechanical system may be transparently composed from textbook diagrams and user generated sketches. We proposed sketch interaction, a seamless interaction technique between sketched content and composed 3D models. The user may use the camera to manipulate models, position, scale, and rotate them, as well as modify the model sketch to apply structural changes on the 3D model. In addition, physical properties can be assigned and modified by sketching while the simulation is running. This is a first step toward flexible 3D content authoring by sketching.

Freehand sketching has to be bounded to a specific domain to allow robust interpretation, especially when using a regular pencil and a low cost camera for acquiring the sketch. We demonstrated that sketch interpretation and geometric reconstruction techniques allow the composition of mechanical systems. These abilities can encourage students to experiment with physical simulations and learn about physics in a playful way, using sketching as a means for communication.

In the future, we would like to utilize the proposed system to explore how intuitive it is for users to sketch 3D models and interact with them by sketching. We would like to test sketch interaction for physical systems in particular. Finally, we would like to extend the system to support a richer physics language.

The setup required for our application consists of a simple camera, one button, a screen, and a CPU. We plan to port the application to cellular phones, which will encourage not only students to experiment with physics, but anyone carrying a mobile phone.

Adopting concepts similar to those suggested in this work will unleash the ability to sketch anywhere from a whiteboard to a napkin, and to interact with the result in a natural and fun way.

7 ACKNOWLEDGMENTS

This work was supported by the Lynn and William Frankel Center for Computer Sciences and by the Tuman Fund. We would like to thank Dr. Peter Ashley Clifford Varley for the help in performing geometrical reconstruction. We would like to thank the reviewers for their suggestions, which helped improving the manuscript.

REFERENCES

[1] Ivan Sutherland E.,"Sketchpad: A Man-Machine Graphical Communication System", Technical Report No. 296, Lincoln Laboratory, Massachusetts Institute of Technology via Defense Technical Information Center, Jan. 1963.

[2] Nate Hagbi, Oriel Bergig, Jihad El-Sana, Klara Kedem and Mark Billinghurst, "In-place Augmented Reality", In *7th IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR),* pp. 135-138, Cambridge, UK, Sept. 2008.

[3] Eric Klopfer, Scot Osterweil, Jennifer Groff and Jason Haas, "The instructional power of digital games social networking simulations and how teachers can leverage them", April 2009, http://education.mit.edu/papers/GamesSimsSocNets_EdArcade.pdf.

[4] Kaufmann Hannes and Meyer Bernd, "Simulating Educational Physical Experiments in Augmented Reality", In *International Conference on Computer Graphics and Interactive Techniques ACM SIGGRAPH ASIA 2008 educators programme,* Singapore, Dec. 2008.

[5] Kaufmann Hannes and Meyer Bernd, "Mathematics And Geometry Education With Collaborative Augmented Reality", In *International Conference on Computer Graphics and Interactive Techniques ACM SIGGRAPH 2002, Educators program,* pp. 37-41, San Antonio, USA, Jul. 2002.

[6] Peter Fiala and Nicoletta Adamo V., "ARpm: an augmented reality interface for polygonal modeling", In *Fourth IEEE and ACM International Symposium on Mixed and Augmented Reality (ISMAR),* pp. 196-197, Vienna, Austria, Oct. 2005.

[7] BuildAR, http://www.hitlabnz.org/wiki/BuildAR.

[8] A. Henrysson, M. Billinghurst and M. Ollila, "Virtual Object Manipulation using a Mobile Phone", In *the 15th International Conference on Artificial Reality and Telexistence (ICAT 2005),* pp. 164-171, Christchurch, New Zealand, Dec. 2005.

[9] Randall Davis,"Magic Paper: Sketch-Understanding Research", *Computer.,* vol. 40, no. 9, pp. 34-41, Aug. 2007.

[10] Christine Alvarado and Randall Davis, "Resolving Ambiguities to Create a Natural Sketch-Based Interface", In *Seventeenth International Joint Conference on Artificial Intelligence (IJCAI),* pp. 1365-1371, Seattle, USA, Aug. 2001.

[11] Burak Levent Kara, Gennari Leslie and F. Thomas Stahovich, "A sketch-based interface for the design and analysis of simple vibratory mechanical systems", In *ASME International Design Engineering Technical Conferences,* Utah, USA, Sept. 2004.

[12] Burak Levent Kara and F. Thomas Stahovich, "An Image-Based, Trainable Symbol Recognizer for Hand-drawn Sketches", *Computers & Graphics.,* vol. 29, no. 4, pp. 501-517, 2005.

[13] Kurtoglu Tolga and F. Thomas Stahovich, "Interpreting Schematic Sketches Using Physical Reasoning", In *AAAI Spring Symposium, Sketch Understanding.* pp. 78-85, Palo Alto, USA, Mar. 2002.

[14] P. Farrugia J., J. Borg C., K. Camilleri P., C. Spiteri and A. Bartolo, "A Cameraphone-Based Approach for the Generation of 3D Models from Paper Sketches", In *Eurographics Workshop on Sketch-based Interfaces and Modelling,* pp. 32-42, Grenoble, France, Aug. 2004.

[15] Pedro Company, Ana Vicent Piquer, Manuel Contero and Ferran Naya, "A survey on geometrical reconstruction as a core technology to sketch-based modeling", *Computer and Graphics.,* vol. 29, no. 6, pp. 892-904, Dec. 2005.

[16] Cao Liangliang, Liu Jianzhuang and Tang Xiaoou, "What the Back of the Object Looks Like: 3D Reconstruction from Line Drawings without Hidden Lines", *IEEE Transactions on Pattern Analysis and Machine Intelligence.,* vol. 30, no. 3, pp. 507-517, March 2008.

[17] Lipson, H. "Reconstruction of a 3D object from a single freehand sketch as means for CAD interface for conceptual design and analysis", Ph.D. Thesis, The Technion Israel Institute of Technology, 1998.

[18] hod Lipson and M. Shpitalni, "Optimization-based reconstruction of a 3D object from a single freehand line drawing", *Journal of Computer-Aided Design.,* vol. 28, no. 8, pp. 651-663, 1996.

[19] Varley, P. A. C. and Martin, R., R. "Automatic Creation of Boundary-Representation Models from Single Line Drawings", Ph.D. Thesis, University of Wales College of Cardiff, 2002.

[20] P. A. C. Varley and R. Martin R., "A system for constructing boundary representation solid models from a two dimensional sketch - frontal geometry and sketch categorisation", In *Proceedings of the first Korea–UK joint workshop on geometric modeling and computer graphics,* pp. 113-128, Seoul, Korea, Apr. 2000.

[21] P. A. C. Varley and R. Martin R., "A system for constructing boundary representation solid models from a twodimensional sketch - topology of hidden parts", In *Proceedings of the first Korea–UK joint workshop on geometric modeling and computer graphics,* pp. 129-144, Seoul, Korea, Apr. 2000.

[22] P. A. C. Varley and R. Martin R., "A system for constructing boundary representation solid models from a twodimensional sketch - geometric finishing", In *Proceedings of the first Korea–UK joint workshop on geometric modeling and computer graphics,* pp. 145-158, Seoul, Korea, Apr. 2000.

[23] Serge Belongie, Jitendra Malik and Jan Puzicha, "Shape Matching and Object Recognition Using Shape Contexts", *IEEE Transactions on Pattern Analysis and Machine Intelligence.,* vol. 24, no. 24, pp. 509-522, April 2002.

[24] L. Lam, Seong W. Lee and Ching Y. Suen, "Thinning Methodologies-A Comprehensive Survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence.,* vol. 14, no. 9, pp. 879, September 1992.

[25] Open Dynamics Engine, http://www.ode.org/.

[26] Tipler, A., Paul and Mosca, G. *Physics for Scientists and Engineers.* 5th. W. H. Freeman, 2003.

[27] Intel OpenCV, http://opencvlibrary.sourceforge.net/.

[28] Kato, H., Billinghurst,M.,"Marker Tracking and HMD Calibration for a video-based Augmented Reality Conferencing System", In *2nd International Workshop on Augmented Reality, IWAR99,* San Francisco, USA., October.

[29] Milgram, P., Kishino,A.F.,"Taxonomy of Mixed Reality Visual Displays", *IEICE Trans. Information and Systems*., vol. E77-D, no. 12, pp. 1321-1329, 1994.