

**The Development of Structural Analysis Virtual Module
for iPad Application**

A Thesis

Presented in Partial Fulfillment of the Requirements for the
Degree of Master of Science
with a
Major in Civil Engineering
in the
College of Graduate Studies
University of Idaho

by

Nicolas L. Peña

December 2014

Major Professor: An Chen, Ph.D., P.E.

Authorization to Submit Thesis

This thesis of Nicolas Peña, submitted for the degree of Master of Science with a Major in Civil Engineering and titled “The Development of Structural Analysis Virtual Module For iPad Application,” has been reviewed in final form. Permission, as indicated by the signatures and dates below, is now granted to submit final copies to the College of Graduate Studies for approval.

Major Professor: _____ Date: _____
An Chen, Ph.D., P.E.

Committee Members: _____ Date: _____
Richard Nielsen, Ph.D., P.E.

_____ Date: _____
Robert Rinker, Ph.D.

Department Chair
of Civil Engineering: _____ Date: _____
Richard Nielsen, Ph.D., P.E.

Dean of the College
of Engineering: _____ Date: _____
Larry Stauffer, Ph.D., P.E.

Final Approval and Acceptance

Dean of the College
of Graduate Studies: _____ Date: _____
Jie Chen, Ph.D.

Abstract

Structural Analysis course is the study of structures subjected to load conditions. This course employs engineering mechanics, material science, and applied mathematics to determine structural deformation, internal forces, and structural support reactions. Without any lab demonstrations, students taking the course miss the opportunity to observe key visual concepts of structural analysis. The iPad application *iStructure* is introduced as a learning tool for students who are taking the Structural Analysis course. The *iStructure* application will illustrate and teach key concepts of structural analysis. This paper will cover the *iStructure* application's three main structural modules: Beam, Trusses, and Frame Module. The structural analysis methods and computer algorithms used to develop the application are discussed within.

Table of Contents

Authorization to Submit Thesis	ii
Abstract.....	iii
Table of Contents.....	iv
List of Figures.....	vii
List of Tables	viii
Chapter 1: Introduction.....	1
1.0 A Review	1
1.1 Purpose of <i>iStructure</i>	1
1.2 Organization	1
Chapter 2: Literature Review.....	3
2.0 Introduction.....	3
2.1 Computer Application in the Civil Engineering Field.....	3
2.2 Educational Mobile Application	3
2.3 Conclusion	4
Chapter 3: Overview of Objective Oriented Programming	5
3.0 Introduction.....	5
3.1 Objective Oriented Programming.....	5
3.2 Objective-C Programming Language	7
3.3 Application of Objective-C Programming in Structural Engineering	9
Chapter 4: Structural Analysis Methodology	10
4.0 Introduction.....	10
4.1 Matrix Structural Analysis.....	10
4.1.1 Introduction.....	10
4.1.2 Direct Stiffness Method.....	10

4.1.3	Post-Processing.....	11
4.2	Structural Deformation	11
4.3	Axial, Shear, and Bending Moment Diagrams	12
4.3.1	Introduction.....	12
4.3.2	Internal Force Diagrams	12
4.3.3	Inflection Points.....	13
4.4	Stress and Strain	13
4.5	Conclusion	14
Chapter 5: Application of Structural Analysis and Computer Algorithms.....		15
5.0	Introduction.....	15
5.1	Truss and Frame Modules.....	15
5.2	Matrix Structural Analysis.....	18
5.2.2	Deflection and Reactions	22
5.2.3	Axial, Shear, and Bending Moment Diagram.....	24
5.2.4	Stress and Strain.....	26
5.3	Summary.....	28
Chapter 6: Alternative Methods for Structural Analysis and Computer Algorithms		29
6.0	Introduction.....	29
6.1	Alternative Methods	29
6.1.1	Beam Deflection, Shear, and Moment Diagrams	29
6.1.2	Continuous Beam.....	29
6.1.3	Influence Line	29
6.2	Computer Algorithms	30
6.2.1	Beam Module.....	30
6.2.2	Deflection and Reactions	31
6.2.3	Shear and Bending Moment Diagram.....	33

6.2.4	Influence Line	33
6.3	Summary	36
Chapter 7: Application Verification		37
7.0	Introduction.....	37
7.1	Beam Structure Module Verification.....	37
7.1.1	Structure Deformation and Reactions.....	37
7.1.2	Shear and Bending Moment Diagram.....	40
7.1.3	Beam Influence Line.....	44
7.1.4	Stress and Strain Diagram.....	48
7.2	Truss Module Verification.....	49
7.2.1	Structure Deformation and Reaction	49
7.2.2	Axial Force Diagram	49
7.2.3	Stress and Strain Diagram.....	52
7.3	Frame Structure Module Verification.....	54
7.3.1	Structure Deformation and Reaction	54
7.3.2	Axial, Shear, and Bending Moment Diagram.....	54
7.3.3	Stress and Strain Diagram.....	62
Chapter 8: Summary and Conclusions.....		64
8.0	Summary	64
8.1	Conclusions.....	64
8.2	Recommendations.....	65
References.....		66
Appendix A: Xcode Object Classes.....		67
Appendix B: iStructure Object Classes		70
Appendix C: iStructure Functions		74

List of Figures

Figure 1: Class Object Example	6
Figure 2: Object Instance Example.....	6
Figure 3: Class Object Hierarchy Example	7
Figure 4: NSObject Hierarchy	7
Figure 5: <i>iStructure</i> 's Custom Load Class Object.....	8
Figure 6: C-Style Array Syntax	8
Figure 7: Run Time Cycle	9
Figure 8: Beam Shape for 4-DOF Element, (Kassimali, 2012).....	11
Figure 9: Internal forces of a 6-DOF element.....	12
Figure 10: <i>iStructure</i> Global Flow Chart.....	15
Figure 11: Frame and Truss Modules Flow Chart.....	16
Figure 12: Matrix Structural Analysis Flow Chart	19
Figure 13: JointMatrix	20
Figure 14: Partitioned Stiffness Pointer Method	21
Figure 15: DrawRec and Reaction Flow Chart.....	23
Figure 16: Axial, Shear, and Bending Moment Flow Chart.....	25
Figure 17: Shear and Bending Moment Due to Applied Loads.....	26
Figure 18: Stress and Strain Flow Chart.....	27
Figure 19: Reaction Influence Line	30
Figure 20: Beam Module Flow Chart.....	31
Figure 21: Beam Analysis Flow Chart	32
Figure 22: Shear and Bending Moment Flow Chart.....	34
Figure 23: Influence Line Flow Chart	35

List of Tables

Table 1: User Objects	17
Table 2: StructureView Object	17
Table 3: Load Object States	18
Table 4: Beam Deflection and Reaction with SAP2000 Results	38
Table 5: Beam Deflection and Reaction with SAP2000 Results for Two-Span Beam	39
Table 6: Beam Shear and Moment Diagram with SAP2000 Results	40
Table 7: Beam Shear and Moment Diagram with SAP2000 Results for Fixed Connections.....	41
Table 8: Beam Shear and Moment Diagram with SAP2000 Results	42
Table 9: Beam Shear and Moment Diagram with SAP2000 Results for Continuous Beam	43
Table 10: Beam Influence Line Animation with SAP2000 Results	44
Table 11: Influence Line for Support Reaction with SAP2000 Results	45
Table 12: Influence Line for Shear Force with SAP2000 Results	46
Table 13: Influence Line for Bending Moment with SAP2000 Results	47
Table 14: Beam Stress and Strain Diagram with SAP2000 Results	48
Table 15: Truss Deflection and Reaction with SAP2000 Results	50
Table 16: Truss Axial Diagram with SAP2000 Results	51
Table 17: Truss Stress and Strain Diagram Validation with Axial Diagram.....	52
Table 18: Truss Stress and Strain Diagram with SAP2000 Results	53
Table 19: Frame Deflection and Reaction with SAP2000 Results	55
Table 20: Frame Axial Diagram with SAP2000 Results for Center Point Loads.....	56
Table 21: Frame Axial Diagram with SAP2000 Results for Lateral Loads	57
Table 22: Frame Shear Diagram with SAP2000 Results for Center Point Loads	58
Table 23: Frame Shear Diagram with SAP2000 Results for Lateral Loads	59
Table 24: Frame Moment Diagram with SAP2000 Results for Center Point Loads.....	60
Table 25: Frame Moment Diagram with SAP2000 Results for Lateral Loads.....	61
Table 26: Frame Stress and Strain Validation with Moment Diagram.....	62
Table 27: Frame Stress and Strain Validations.....	63

Chapter 1: Introduction

1.0 A Review

Structural Analysis is an introductory course for structural engineering, which is taught in most undergraduate civil engineering program. Structural Analysis incorporates the fields of applied mechanics, materials science, and applied mathematics to compute structure characteristics such as: deformations, internal forces, stress/strain, and support reactions under prescribed load and/or other external effects.

iStructure is an iOS application designed for the iPad with the main purpose of providing a visual lab for the Structural Analysis course. With iStructure, students can create and alter structural conditions and apply various loads to visually observe structural deflection, bending, shear and axial characteristics, and reactions on the structure. The application can be used to teach, demonstrate structural analysis concepts, and assist engineering students with a user friendly environment.

This paper will cover the development of the *iStructure* application. The implementation of the structural analysis methods, such as matrix structural analysis, using object-oriented programming will be presented.

1.1 Purpose of *iStructure*

The purpose of iStructure is to incorporate the fundamental and visual aspects of structural analysis into an iPad application and develop a cost-effective cyber-teaching lab for Structural Analysis for engineering students. *iStructure* provides three main modules: Beam, Frame, and Truss. These modules provide an environment where students can simulate, explore, and learn the structural concepts. The structural concepts such as structural deformation, support reactions, internal force diagrams, and stress/strain diagrams are illustrated within each module.

1.2 Organization

Chapter 2 provides a review of studies performed on applications implemented in the Civil Engineering field and studies conducted on educational mobile application. An overview of Objective-C programming and additional information is provided in Chapter 3. Chapter 4 describes the methodologies used for structural analysis for the Truss and Frame modules.

The implementations of the matrix structural analysis method in the iStructure are discussed in Chapter 5, these include the coding and programming needed to develop a complete user interaction iPad application with animated structural behaviors. Chapter 6 describes the alternative structural analysis methods for the Beam Module. In Chapter 7, the *iStructure*'s structural analysis results and visual characteristics are assessed and verified. Lastly, Chapter 8 provides a summary and conclusion of the thesis.

Chapter 2: Literature Review

2.0 Introduction

The purpose of this study is to develop an interactive structural analysis teaching tool application using modern technology, such as the iPad device. The following literature review focuses on the computer application used in the civil engineering field and the use of educational mobile application.

2.1 Computer Application in the Civil Engineering Field

Studies have been done on the application of modern technology in the field of the civil engineering education. At Graz University of Technology in Austria, a study was conducted using multimedia and internet based applications, called e-Learning, in higher education (Ebner & Walder, 2008). The study performed by Ebner, shows that higher education is strongly influenced by modern technology and that educational institute should adapt to meet the current technological generation.

A previous study by Ebner confirmed the success of user-centered game based learning applications for higher education (Ebner & Holzinger, 2007). This study was conducted on 121 students during a Master's level Structural Concrete lecture. The purpose of Ebner's study was to gain insight into whether online games have a positive influence on the student's learning in higher education. The study revealed that it was necessary to create a motivating and enjoyable software application in order to maintain the student's involvement and that modern applications can be useful methods for teaching and education for civil engineering students at Master's level.

2.2 Educational Mobile Application

Cochrane from Unitec in New Zealand, implemented a mobile based teaching project called mobile Web 2.0 for Bachelor of Product Design courses (Cochrane, 2010). The purpose of the study was to explore the potential of mobile Web 2.0 to enhance the teaching and learning environments with the focus on mobile blogging. The study showed that students benefited and preferred the mobile-based activities in place of the traditional paper-based activities. The study yielded successful results in students' engagement and flexibility of learning

through the use of mobile Web 2.0. The outcome of the study later led to the integration of mobile Web 2.0 into the Bachelor of Product Design curriculum in 2009.

A study performed by Herrington at the University of Wollongong, investigated the use of mobile technology in higher education (Herrington & Herrington, 2007). The study showed that the affects of mobile technologies and applications have the potential to enable teachers to adapt the teaching methods to meet the current learning environment for the present generation.

2.3 Conclusion

Ebner's studies show that modern teaching technologies greatly influence the student's learning in higher education. In those studies, Ebner expressed the need for inspiring and developing enjoyable teaching application using new technology in order to gain positive results in the student's learning and stated that "Learners of tomorrow will use technologies of tomorrow and the university has to prepare for and adapt to it" (Ebner & Holzinger, 2007).

Cochrane's and Herrington's study shows that mobile-based teaching applications have the potential to enhance students' learning in higher education. Cochrane's study shows that mobile based teaching technique can be enjoyable and can become the students' preferred learning technique over the traditional based teaching.

Chapter 3: Overview of Objective Oriented Programming

3.0 Introduction

iStructure is an iPad application developed using Xcode software, an Apple based integrated development environment (IDE) software with Objective-C as the primary programming language. Two types of programming paradigms are commonly used: Objective Oriented Programming (OOP) and Procedural Programming (PP). The significant differences between these paradigms are the ability to create and allocate objects in OOP, whereas procedural programming follows a linear step-by-step coding procedure. Examples of procedural programming are seen in FORTRAN applications.

3.1 Objective Oriented Programming

In OOP, programming involves the usage of class objects. A class object is a group of definitions and methods used to define a memory allocated instance. Class object definitions are properties retained by an object instance, which can include other objects instances. Class object methods are private functions that can only be called by the object instance. When class objects are used to define a memory allocated instance, that instance becomes an object instance. An object instance is the physical aspect of a class object and contains all the properties and methods defined by the class object. A class object can be used to create multiple object instances, all having the same definitions and methods but each containing independent values. Figure 1 is an example of a class object Person. A person can have different abilities, such as reading and working, which can be considered the object's methods. In Apple Developer documents, OOP is metaphorically defined as an "actor" that has "human-like intentions and abilities" (Apple Inc, 2010).

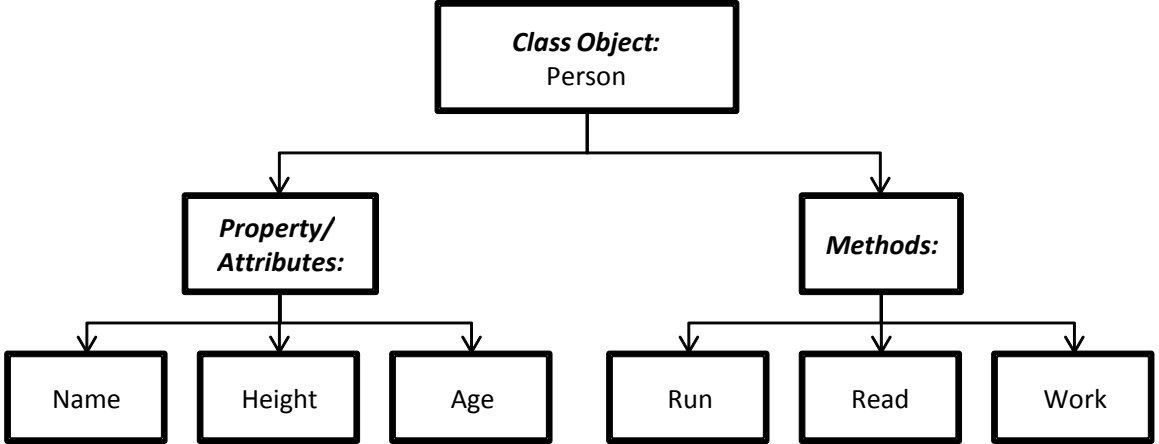


Figure 1: Class Object Example

Figure 2 is an example of multiple object instances, the first two having the same object class and last having a different object class. Mark and Sally are object instances of the Person class object, each having their individual property values. Alternatively, Fido is an object instance of the Dog class object which consists of different properties and methods than that of the Person class object.

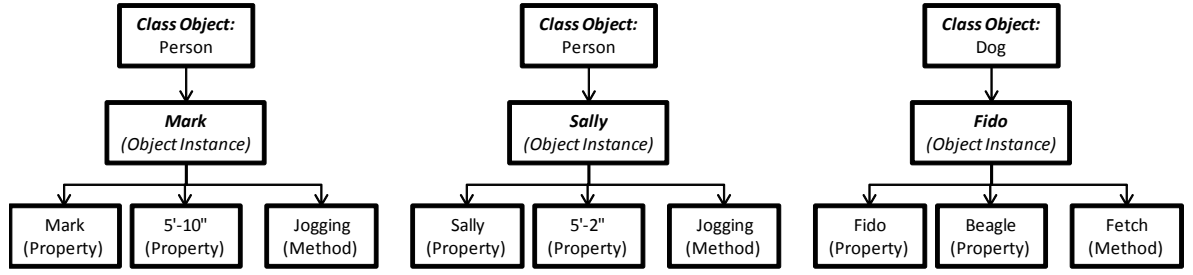


Figure 2: Object Instance Example

Class objects form a hierarchy tree, where child classes inherit parent classes' definitions and methods. Figure 3 is an example of a hierarchy tree where the class object Organism is the root parent class and all the sub-class objects are child classes. Thus, the child class object Person inherits the all the properties of the Human class object, which the Human class object inherits from the Organism class object. As a result, the Person class object inherits all the properties and methods up to the root class object.

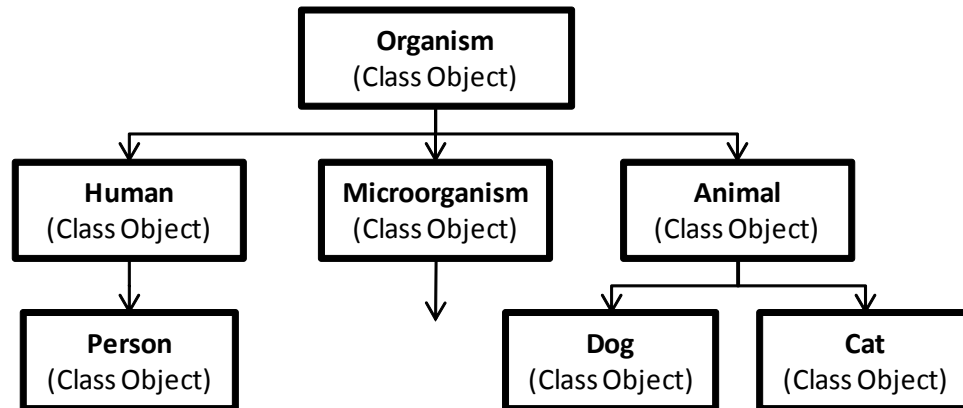


Figure 3: Class Object Hierarchy Example

3.2 Objective-C Programming Language

In Xcode, the root class object is NSObject, and the UIObjects (i.e. UIResponder, UIView, and etc.) are descendant classes, as shown in Figure 4.

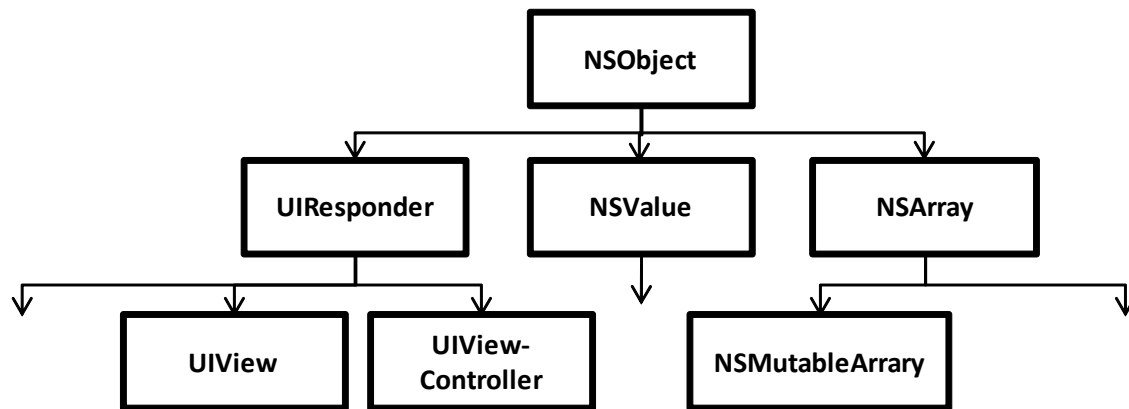


Figure 4: NSObject Hierarchy

Objective-C programming allows the developers the ability to create custom object classes and methods. A custom class is created from a parent class and inherits the parent's definitions and methods along with the new definitions and methods. Custom class objects can be used to manage a group of associated properties. The custom class object's methods can be used to make private functions and/or routines, which are custom to the object instance. In the *iStructure* application, custom load class objects were created to define the general properties that all load objects share. From the Load class object, special classes were

then created to define the special properties that each load type shared. These child classes consist of PointLoad, PumpLoad, and Support class objects. Figure 5 is an example of the custom LoadObject class object used in *iStructure* and the class object's child classes. The LoadObject is a custom object class inherited from the NSObject class and the child classes are custom object classes inherited from the LoadObject class object. Thus the child classes contain the LoadObject definitions and new class definitions.

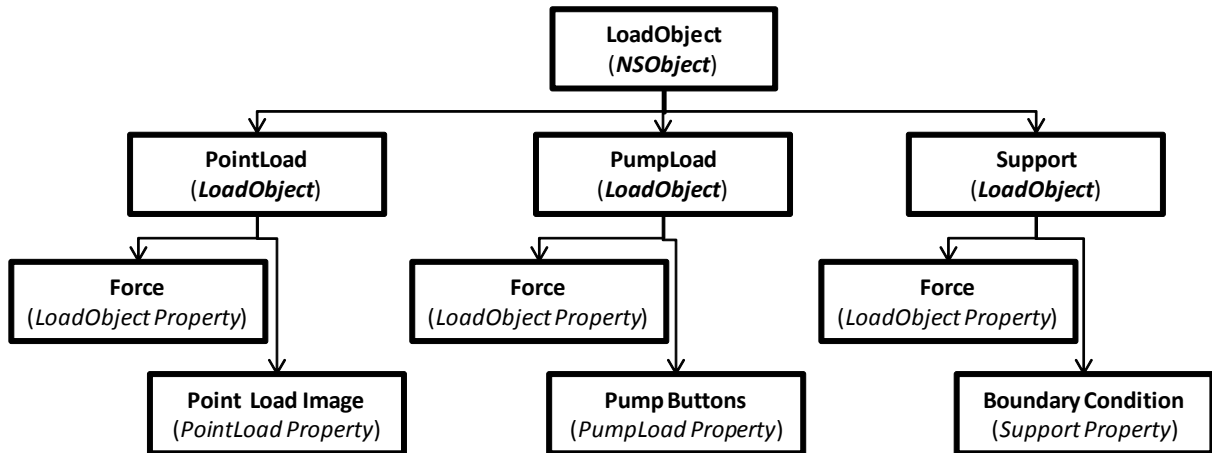


Figure 5: *iStructure's* Custom Load Class Object

One of the advantages of using Objective-C programming is the ability to use C-style array syntax. Figure 6 is an example of a C-style array code. The benefits of using C-style array are the simplicity and relative ease of coding. Most importantly C-style array are variables that do not need to allocate memory, which reduces the application's memory demand.

```

function main() {
    var array1 array[250] integer
    var array2 array[] integer
    var array3 array[] integer
    var i integer
    for(i=0;i<250;i=i+1) {
        array1[i]=i
    }
}
  
```

Figure 6: C-Style Array Syntax

Since C-style array variables are not retained, they cannot be easily sent to other main files. In this application, the computation is performed using C-style array syntax and at the end of the routine the C-style arrays are stored in memory allocated objects called NSMutableArray.

3.3 Application of Objective-C Programming in Structural Engineering

Objective-C programming plays an important role in the development of the *iStructure*. Objective-C programming has a substantial library of user-interaction oriented objects and methods, which allow the application to be more user-oriented.

Aside from the user-oriented capability, Objective-C programming provides data management which is crucial when an application is continuously receiving and processing large amounts of information. This feature can provide real-time reactions to user interactions and provide a series of animated results, which can provide a unique and visually stimulating simulation. Figure 7 illustrates an example of a Run Time cycle for Main file.

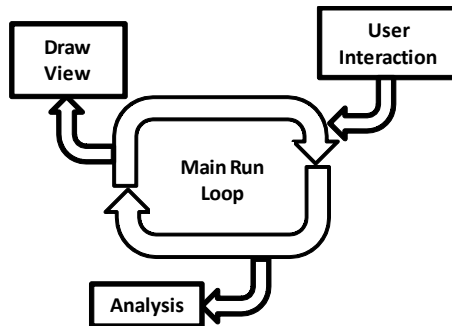


Figure 7: Run Time Cycle

The main run loop allows the user to interact and alter the function analysis instantly and output the results in a form of a visual solution. This feature gives the ability to simulate custom animation, such as objects falling and animated structural deformation.

Chapter 4: Structural Analysis Methodology

4.0 Introduction

Matrix structural analysis is a common method of solving indeterminate structures. Matrix structural analysis can be applied to a structural system in order to determine the structure's nodal reactions and displacements using the system's stiffness. This chapter will review the theory of matrix structural analysis method used in the *iStructure* application. Direct stiffness method, structural deformation, and more will be reviewed.

4.1 Matrix Structural Analysis

4.1.1 Introduction

There are two approaches for matrix structural analysis: flexibility and direct stiffness methods. The flexibility method uses redundant forces as primary unknowns, which are then determined using the structure's compatibility equations. Once the redundant forces are known, the displacement can then be calculated using the structure's equilibrium equations. In the direct stiffness method, the primary unknowns are the nodal displacements which are determined using the structure's equilibrium equation. Once the nodal displacements are known, the redundant forces can then be determined through the structure's compatibility equations (Kassimali, 2012). The direct stiffness method will be employed for the structural analysis.

4.1.2 Direct Stiffness Method

The direct stiffness method implements the structure's stiffness relationship for determining the structure's displacement and reactions. This relationship can be expressed in Eq. (1).

$$F + F_e = Kd \quad (1)$$

$$F = \begin{bmatrix} R \\ f \end{bmatrix}, \quad F_e = \begin{bmatrix} F_{e_c} \\ F_{e_f} \end{bmatrix}, \quad K = \begin{bmatrix} K_{cc} & K_{cf} \\ K_{fc} & K_{ff} \end{bmatrix}, \quad d = \begin{bmatrix} d_c \\ d_f \end{bmatrix} \quad (2)$$

Where F is the external loads applied to the structure, F_e is the equivalent nodal loads. Within F and F_e vector, R and F_{e_c} represent the forces at constrained Degree of Freedoms (DOFs), while f and F_{e_f} represents the forces at the unconstrained or free DOFs. K represents the structure's stiffness matrix with subscripts c and s representing constrained and

unconstrained DOFs, respectively. \mathbf{d} is the structure's nodal displacement. External loads applied to the structure's members are converted into equivalent nodal loads, \mathbf{F}_e . Once the external loads and structure's stiffness are determined, the structure's nodal displacement can be determined. With the displacements known, the reaction forces can then be determined using Eq. (3).

$$\mathbf{R} = \mathbf{K}_{cc}\mathbf{d}_c + \mathbf{K}_{cf}\mathbf{d}_f - \mathbf{F}_{ec} \quad (3)$$

4.1.3 Post-Processing

Once the structural system's deformation is determined, internal forces of the system including axial, shear, and bending forces can then be determined. Further processing of the internal forces or reactions can provide information about the structure's compatibility, stress and strain conditions. Next the structural deformation, internal forces, and stress and strain will be reviewed.

4.2 Structural Deformation

The structural deformation is represented by the nodal displacement and structural member's deformation. The nodal displacement is determined from the direct stiffness method, while the beam shape function is used to determine the structural member's deformation, as shown in Figure 8.

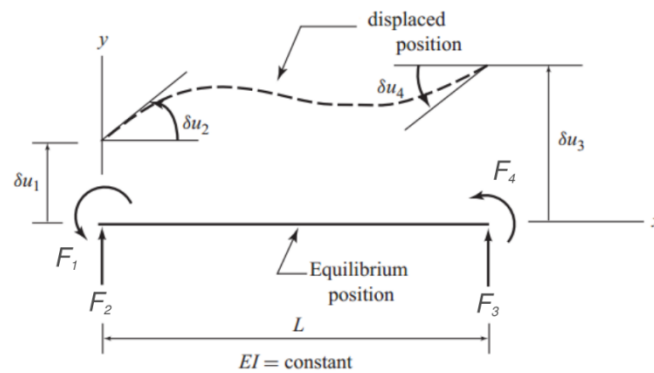


Figure 8: Beam Shape for 4-DOF Element, (Kassimali, 2012)

The beam shape function (Kassimali, 2012) is defined in Eq. (4), where \mathbf{N}_i is the member shape-function matrix and \mathbf{u}_i is the nodal displacement (δu), shown in Figure 8.

$$u_y = \mathbf{N}_i \mathbf{u}_i \quad (4)$$

The member shape-function matrix is expressed in Eq. (5), where x is a point along the length (L) of the member.

$$N = \begin{bmatrix} 1 - 3\left(\frac{x}{L}\right)^2 + 2\left(\frac{x}{L}\right)^3 \\ x\left(1 - \frac{x}{L}\right)^2 \\ 3\left(\frac{x}{L}\right)^2 - 2\left(\frac{x}{L}\right)^3 \\ \frac{x^2}{L}\left(-1 + \frac{x}{L}\right) \end{bmatrix} \quad (5)$$

4.3 Axial, Shear, and Bending Moment Diagrams

4.3.1 Introduction

The axial, shear, and bending moment diagrams illustrate the structure's internal forces experienced due to the various loading conditions. The axial force diagram displays the axial forces induced in structure members in a given structural configuration, while the shear force diagram displays the internal shear forces, and the bending moment force diagram displays the internal bending moments. These diagrams are structural tools vital for the understanding of the behavior of the structural systems.

4.3.2 Internal Force Diagrams

For a frame element with three Degree of Freedoms (DOFs) per node, the local nodal forces consist of axial, shear, and bending moment at each node. Figure 9 is an example of the local nodal forces (N -axial, V -shear, and M -moment) in a 6-DOF node element with non-nodal external forces (F_N , F_V , and F_M) applied. For a truss element with 2-DOS per node in a global coordinate system, the local nodal forces typically consist of axial forces.

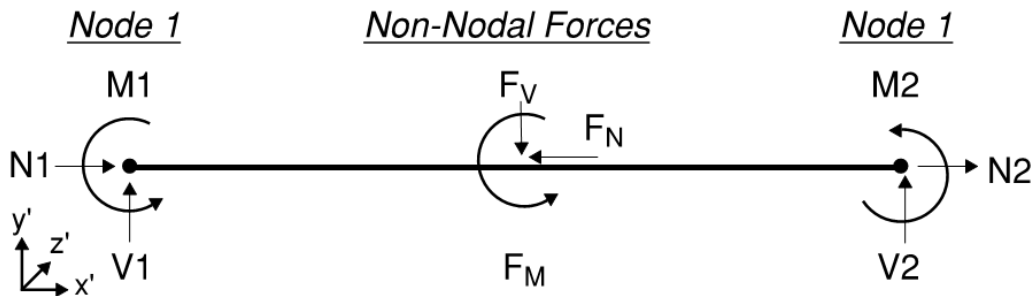


Figure 9: Internal forces of a 6-DOF element

The internal forces are obtained by post-processing the results of the matrix structural analysis and transforming the resulting global nodal forces into local nodal forces. The global coordinate system is made up of the coordinates in which the structural system is defined, and the local coordinate system consists of the coordinates at a structural element, shown in Figure 9. The relationship between the global and local coordinates is defined by the transformation matrix.

The non-nodal external forces (F_N , F_V , and F_M), in Figure 9, must be included in the development of the internal force diagrams. By including axial, shear, bending moment forces, and non-nodal forces, the respective diagrams can be developed and illustrated for any given structural system.

4.3.3 Inflection Points

Inflection points are points of transition between concave and convex curvatures in an element. Given the relationship between curvature and bending moment, inflection points are transition points between positive and negative bending moments.

4.4 Stress and Strain

The relationship between stress and strain is described by the constitutive equation, and is expressed in Eq. (6) for linear elastic material.

$$\sigma = E\varepsilon \quad (6)$$

Where σ is stress, ε is strain, and E is the modulus of elasticity and represents the relationship between the stress and strain of a material. In a 3-DOF element, strain in the local x-direction is the summation of axial and bending strains, as expressed in Eq. (7). In a 2-DOF element, axial forces are generated internally which results in only axial stress and strain.

$$\varepsilon = \varepsilon_{bending} + \varepsilon_{axial} \quad (7)$$

Bending strain is strain resulting from the element's curvature along the element's profile, expressed in Eq. (8).

$$\varepsilon_{bending} = -\kappa y \quad (8)$$

Where κ is the beam curvature and y is the vertical distance from the beam's neutral axis. Approximately, the curvature can be defined by Euler Bernoulli's Beam Theory, expressed in Eq. (9).

$$\kappa \approx \frac{d^2y}{dx^2} = \frac{M(x)}{EI} \quad (9)$$

Axial strain is strain resulting from the element's elongation or shortening. The axial strain is expressed in Eq. (10).

$$\varepsilon_{axial} = \frac{dL}{L} \quad (10)$$

4.5 Conclusion

Matrix structural analysis can be applied to a structural system in order to determine the structure's characteristics. The direct stiffness method can be used to determine the structure's deformation and reactions. The structural deformation can be used to illustrate the structure's nodal and member deformation. Lastly, the internal force and stress and strain diagram can be used to display the structure's internal forces and stress & strain behavior.

Chapter 5: Application of Structural Analysis and Computer Algorithms

5.0 Introduction

The theories in Chapter 4 are implemented in the *iStructure* application and are described in this chapter. The *iStructure* application is comprised of three modules: Beam, Frame, and Truss modules. Sub-functions for structural deformation, moment and shear diagram, and stress and strain diagram are developed within each module, as shown in Figure 10.

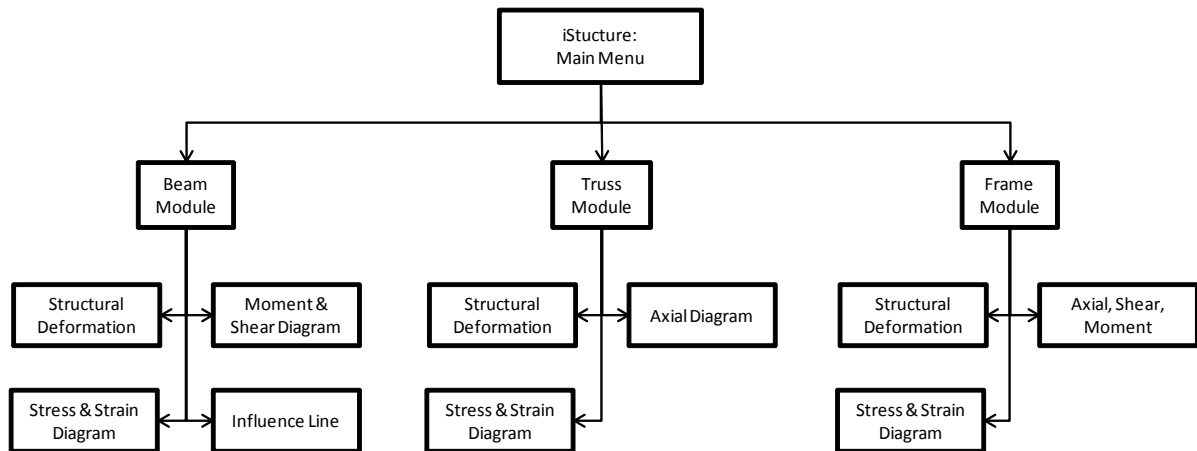


Figure 10: *iStructure* Global Flow Chart

The Frame and Truss modules utilize the matrix structural analysis methods for determining structural deformations and reactions. The beam module incorporates alternative methods, which will be discussed in the next section. Details for the Frame and Truss modules' functions and matrix structural analysis method implementation will be provided in this section.

5.1 Truss and Frame Modules

The Truss and Frame modules consist of two individual UIViewController, as shown in Appendix A. The UIViewController acts as the main and header file for the Frame and Truss modules and is where the corresponding sub-functions are called from the Frame and Truss modules. The Frame and Truss modules share the same procedures with minor differences, which are defined in Section 4.1. The flow chart in Figure 11 describes the processes for the Frame and Truss modules.

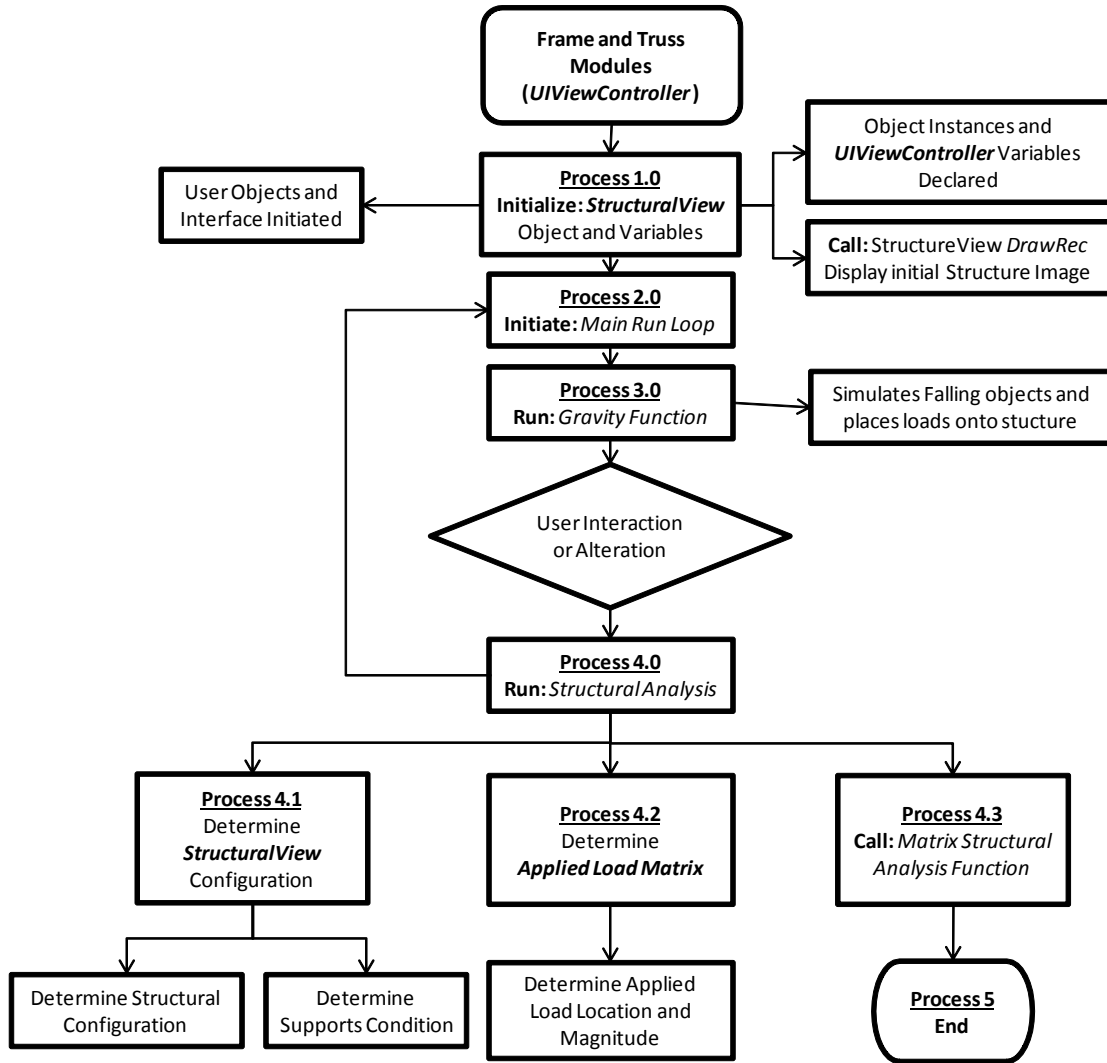


Figure 11: Frame and Truss Modules Flow Chart

5.1.1.1 Frame and Truss Modules: Process 1.0

During the first stage of the Frame and Truss modules, Process 1.0 initializes all user and view objects and displays these objects in the application's interface. Table 1 lists common user objects that are initialized and the objects' properties. Within this process, the StructureView object instance is created and allocated. StructureView is a UIView custom class representing the structure being analyzed, as shown in Table 2 and Appendix B. During the initialization of the StructureView object, the object's properties are firstly assigned, and then the StructureView's *DrawRect* function is called for to draw the initial structure's view (see Appendix C for details on DrawRect function), which is the un-deformed shape of the structure.

Table 1: User Objects

User Object		
Object Class	Object Parent Class	Properties
LoadObject	<i>NSObject</i>	<i>UIImage, Magnitude, Location, etc.</i>
PumpObject	<i>NSObject</i>	<i>UIImage, UIButton, Magnitude, Location, etc.</i>
SupportObject	<i>NSObject</i>	<i>UIImage, UIMenu, Reaction, Location, etc.</i>

Table 2: StructureView Object

StructureView Object	
Object Parent Class:	<i>UIView Class</i>
Properties:	<i>Structural Properties</i>
	<i>Structural Configuration</i>
	<i>Nodal Deformation Matrix</i>
	<i>Local Forces Matrix</i>

5.1.1.2 Truss and Frame Modules: Process 2.0 and Process 3.0

In Process 2.0, the time-based *Main Run Loop* function is initiated. The *Main Run Loop* function initiates other processes at a time interval. The designated time and Process 3.0 are correlated to provide a smooth simulation of the falling of load objects.

In Process 3.0, the Gravity Function is initiated which is responsible for shifting the falling load objects downward at an incremental distance and is responsible for the load objects' interaction with each other. Table 3 lists different load object states, which determine how load objects behave and react. Once a load object is considered falling, the object will continue to move downward incrementally until it is out of the user's view or lands on another object. If the object is out of the user's view, the object is deleted and automatically deallocated. If the object lands on another object, its state is changed to contact state and the movement is stopped. During this state the object is interacting with other objects and treated as an applied load. In the Frame and Beam modules the load objects can interact and stack upon each other; which is defined in the Gravity function. But in the Truss module, when two load objects are applied at the same joint, they will be combined into one load object with the total load, since the load objects can only be applied at the joints (nodes) of the structures.

The stacking of loads was not implemented in the Truss module because of the visual interference between the load images and the truss structure image.

Table 3: Load Object States

Load Object States	
<i>Object State</i>	<i>Description</i>
Initial State:	<i>Object remains unmoved and unaffected by gravity</i>
Touching State:	<i>Object moves with User touch and unaffected by gravity</i>
Falling State:	<i>Object is affected by gravity and moves downward</i>
Contact State:	<i>Object is in contact with the structure or other load objects, it remains unmoved, unaffected by gravity, and is added to the applied load matrix</i>

5.1.1.3 Frame and Truss Modules: Process 4.0

Structural analysis is initiated in Process 4.0. The initiation of Process 4.0 is dependent on a structural analysis variable, which is a Boolean variable set to TRUE only during a user interaction or structural alteration that requires the re-analysis of the structure to prevent any unnecessary iterations of structural analysis. During structural analysis, structural configuration, including supporting conditions and structural type, is determined first and the values are assigned to the StructureView object in Process 4.1, as shown in Table 2. In Process 4.2, the load objects that are being applied to the structure are stored in an NSMutableArray object named AppliedLoadMatrix, which is initiated in Process 1.0.

In Process 4.3, the updated StructureView and AppliedLoadMatrix are sent to the *Matrix Structural Analysis* function, which will calculate the nodal displacements and forces and then update to the StructureView object accordingly.

5.2 Matrix Structural Analysis

The *Matrix Structural Analysis* function is defined in an Objective-C Category file (see Appendix A). The objective of the Matrix Structural Analysis function is to calculate the structure's reactions at the supports and joint (nodal) displacements. Figure 12 displays the flowchart for this function.

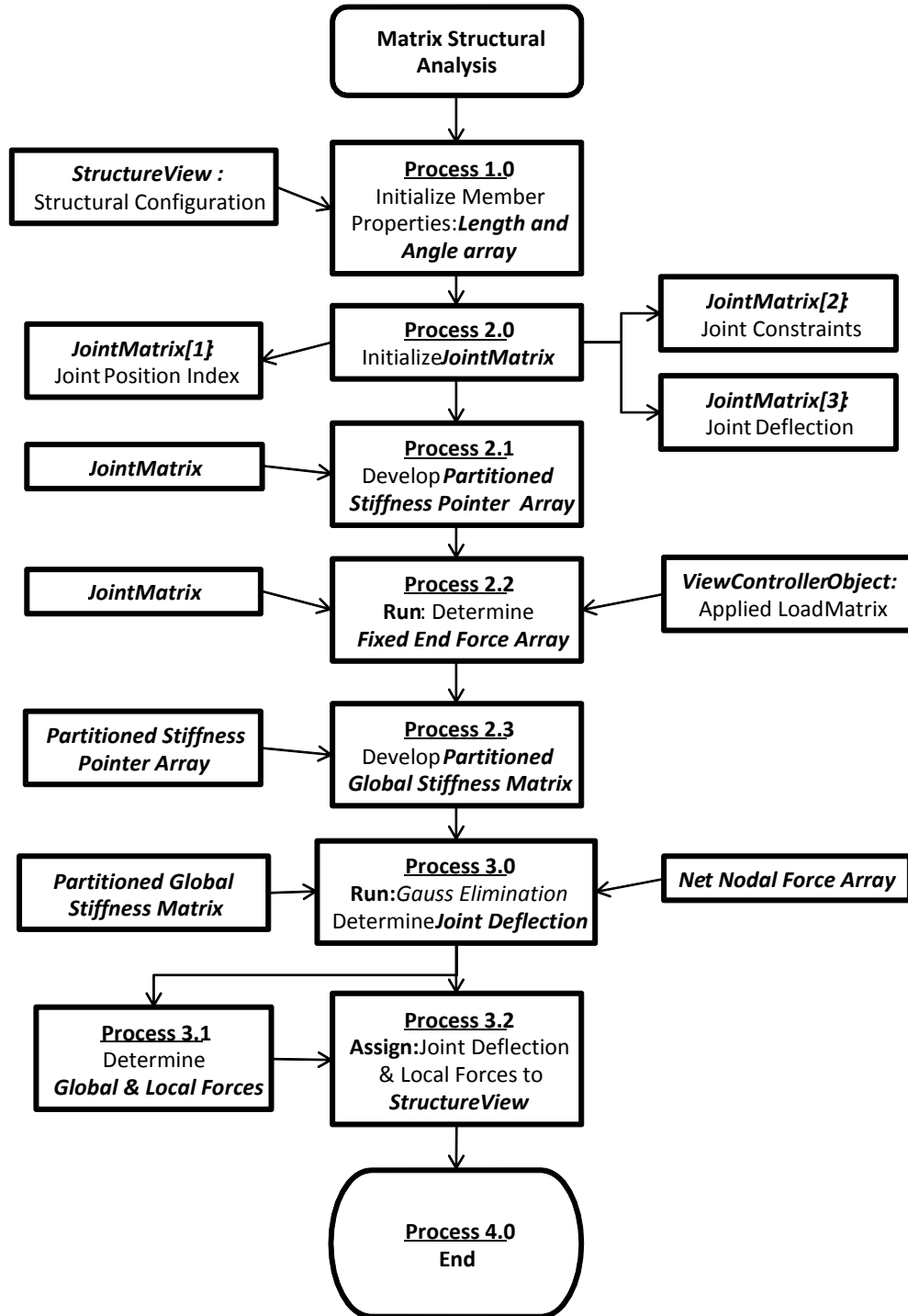


Figure 12: Matrix Structural Analysis Flow Chart

5.2.1.1 Matrix Structural Analysis: Process 1.0 and Process 2.0

All structural members are initialized during Process 1.0. Each structural member has its own properties such as member length, angle, and nodes. Once the structural members are defined, Process 2 is initiated and the JointMatrix is initialized. The JointMatrix is a three dimensional matrix. The first two dimensions represent a standard matrix, while the third dimension contains multiple matrixes. The joint position index matrix, joint constraint matrix, and joint deflection matrix are held in the JointMatrix. The combined matrix facilitates data handling and accessibility. In the application, the same nodal index is used for each matrix; this provides easy accessibility between matrixes within the JointMatrix, as shown in Figure 13.

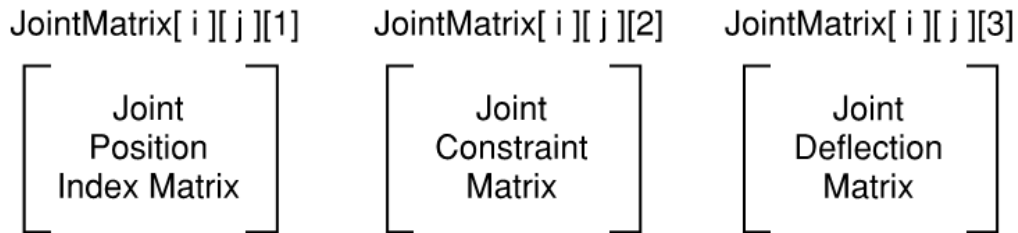


Figure 13: JointMatrix

In Process 2.1, the joint position index and constraint matrices are used to develop a Partitioned Stiffness Pointer array. Instead of developing an M-number (number of members) of local structural stiffness matrices, combining them into an NxN (number of nodes) global stiffness matrix and then partitioning the global stiffness matrix, the Partitioned Stiffness Pointer determines the partitioned nodes and their position in the global and partitioned global indexes so that the partitioned global stiffness matrix can be determined directly. The Partition Global Stiffness Matrix is comprised of the global stiffness coefficients of the unconstrained or free DOFs in the structural system and is used to determine the structure's nodal displacements.

In Process 2.2, the Net Nodal Forces array is developed, as shown in Equation (11), where F and F_e are defined in Section 4.1.2.

$$\text{Net Nodal Forces} = F - F_e \quad (11)$$

In Process 2.3, the Partitioned Global Stiffness Matrix is determined. The Predetermined Global Stiffness Matrix function is used to simplify the computation demand (see Appendix

B). The Predetermined Global Stiffness Matrix function uses an index position argument and returns a single matrix element value, which corresponds to the index position. By using the Partitioned Stiffness Pointer array to retrieve the partitioned nodes' index position, the global stiffness value at the index position can be obtained and stored into the Partitioned Global Stiffness Matrix at the nodes' partitioned global stiffness position, as shown in Figure 14. The indices i & j corresponds to the node's global stiffness matrix position and x & y indices corresponds to the node's partitioned global stiffness matrix position. Both indices are recorded for each node and are stored in the Partitioned Stiffness Pointer array.

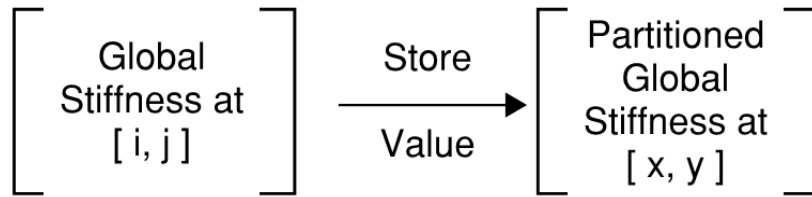


Figure 14: Partitioned Stiffness Pointer Method

5.2.1.2 Matrix Structural Analysis: Process 3.0 and Process 4.0

In Process 3.0, the Partitioned Global Stiffness Matrix and Fixed End Force Array are sent to the *Gauss Elimination* function to calculate the joint (nodal) displacements, as shown in Eq. (12):

$$F_e = Kd \rightarrow \begin{bmatrix} F_{ec} \\ F_{ef} \end{bmatrix} = \begin{bmatrix} K_{cc} & K_{cf} \\ K_{fc} & K_{ff} \end{bmatrix} \begin{bmatrix} d_c \\ d_f \end{bmatrix} \quad (12)$$

Since the F_{ef} forces are known, Eq. (12) can be simplified to Eq. (13).

$$F_{ef} = (F + F_e)_f = K_{fc}d_c + K_{ff}d_f \quad (13)$$

where $(F + F_e)_f$ is the Net Nodal Forces array of unconstrained DOF, K_{fc} is the Partitioned Global Stiffness Matrix of constrained and unconstrained nodes, K_{ff} is the Partitioned Global Stiffness Matrix of unconstrained nodes, d_c is the nodal displacement of constrained nodes, and d_f is the nodal displacement of unconstrained nodes. In this application, prescribed displacements (d_c) are not included in the structural analysis, thus Eq. (13) can be reduced to Eq. (14):

$$(F + F_e)_f = K_{ff}d_f \quad (14)$$

Eq. (14) is sent to the *Gauss Elimination* function and the unconstrained DOF nodal displacements (\mathbf{d}_f) are determined and stored in the StructureView Properties.

In Process 3.1, the support reactions (\mathbf{R}) are determined using the Eq. (15):

$$R = K_{cc}d_c + K_{cf}d_f - F_{e_c} \quad (15)$$

Since the prescribed displacements are not considered, Eq. (15) can be simplified to Eq. (16):

$$R = K_{cf}d_f - F_{e_c} \quad (16)$$

The index positions for the support reactions (\mathbf{R}) are retrieved from the StructureView properties. Once the indices are known, the global stiffness (\mathbf{K}_{cf}), displacement (\mathbf{d}_f), and equivalent nodal forces (\mathbf{F}_{e_c}) can be retrieved and used to calculate the support reactions.

Local forces are then determined by transforming the global forces using the transformation function, as shown in Eq. (17).

$$f = T(\theta) (F + F_e) \quad (17)$$

Where f is the local forces, $T(\theta)$ is the transformation function at angle θ , and $(F + F_e)$ is the net global forces.

In Process 3.2, local and global forces and the nodal deflections are stored in separate NSMutableArray and the StructureView's properties are updated. Once the Matrix Structural Analysis function is completed in Process 4.0, the Deflection and Reaction function is initiated.

5.2.2 Deflection and Reactions

The objective of the Deflection and Reaction function is to display the support reactions and deformed shape of the structure. The functions to assign the reactions are found in the module's UIViewController file, and the structure's deflection function is in the Beam Shape Category file and StructureView file. The flow chart of the Deflection and Reaction function is illustrated in Figure 15

5.2.2.1 Deflection and Reactions: Process 1.0 and Process 2.0

Once the Deflection and Reaction function is initiated, the support reactions obtained from the Matrix Structural Analysis function are assigned to the Support Objects' UILabel, which

displays the support reactions in the user interface. In Process 1.1, the StructureView's *setNeedsDisplay* from the UIViewController is called to initiate the *DrawRect* function. Due to Objective-C protocol, *DrawRect* cannot be directly called and can only be initiated through *setNeedsDisplay* function (see Appendix C).

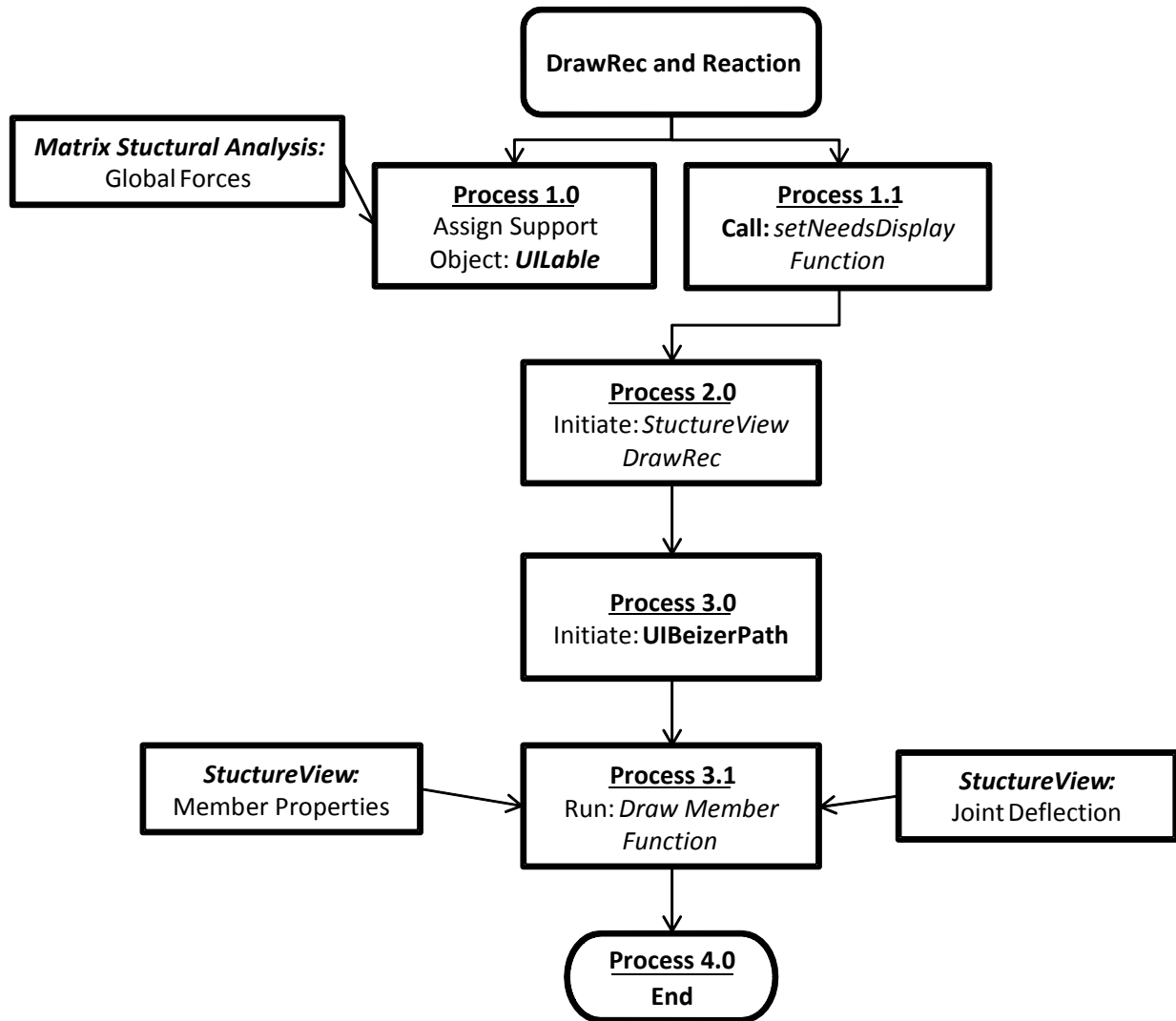


Figure 15: DrawRec and Reaction Flow Chart

5.2.2.2 Deflection and Reactions: Process 3.0 and Process 4.0

In Process 3.0, UIBezierPath object StructureBound and MemberBounds are initiated (see Appendices A and B). In Process 3.1, the *Draw Member Bounds* function is called and iterated for each structural member. In each iteration, the member's nodal displacements are

sent to the *Beam Shape* function and the coordinates of deflection points, which are used to draw the deformed member using the MemberBounds object, are generated and returned to the *Draw Member Bound* function.

Once a MemberBounds shape is completed and closed, it is given a fill and stroke color. Then the MemberBounds is appended to the StructureBound and stored in a MemberBoundMatrix to be use in the StructureView configuration process.

In Process 4.0, the StructureBound is called to draw the deformed structure on the user interface once all the structure's members have been iterated and the MemberBounds shape has been appended to StructureBound.

5.2.3 Axial, Shear, and Bending Moment Diagram

The axial, shear and bending moment diagrams are calculated from the internal forces (local forces) acquired from the Matrix Structural Analysis function. They are drawn within the StructureView file and initiated when *DrawRect* is initiated.

5.2.3.1 Axial, Shear, and Bending Moment: Process 1.0

Once the StructureView's DrawRect is initiated and the structure is drawn, the StructureView's diagram variable is assigned a value, corresponding to either axial, shear, or moment, when "draw diagram" is activated by a user.

5.2.3.2 Axial, Shear, and Bending Moment: Process 2.0, 3.0, and 4.0

Based on the StructureView's diagram variable, which is initiated in Process 1.0, the appropriate diagram function is initiated for either Process 2.0, 3.0, or 4.0. Once a diagram function is initiated, the function iterates for all the structural members and draws the diagram based on the StructureView's Local Forces for each structural member. In Process 3.2 and Process 4.2, the applied loads are sent to sub-function to determine the applied load's shear and moment diagram, as discussed in Section 4.3.2 and shown in Figure 9. Figure 17 gives an example of a shear and bending moment diagram for an applied point load.

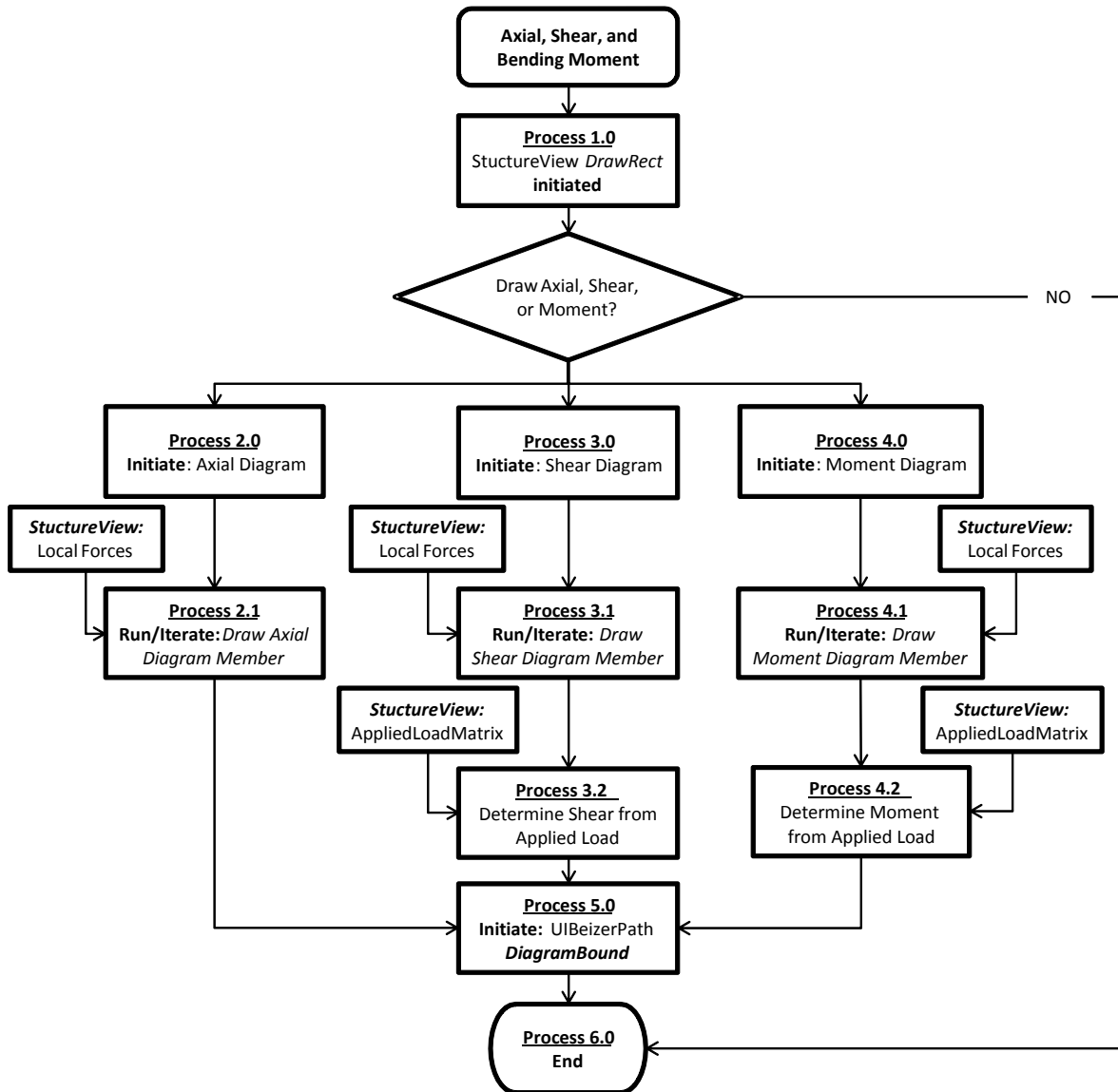


Figure 16: Axial, Shear, and Bending Moment Flow Chart

Since the applied axial forces (F_N) are not considered in the application, an axial applied loads function is not required. The applied loads at the members can be retrieved from the StructureView's AppliedLoadMatrix. Each applied load's diagrams are superimposed into one member diagram, respectively.

5.2.3.3 Axial, Shear, and Bending Moment: Process 5.0

Process 5.0 is initiated and the diagram values are assigned to the UIBezierPath object DiagramBound and drawn on the user interface once the diagram values are determined.

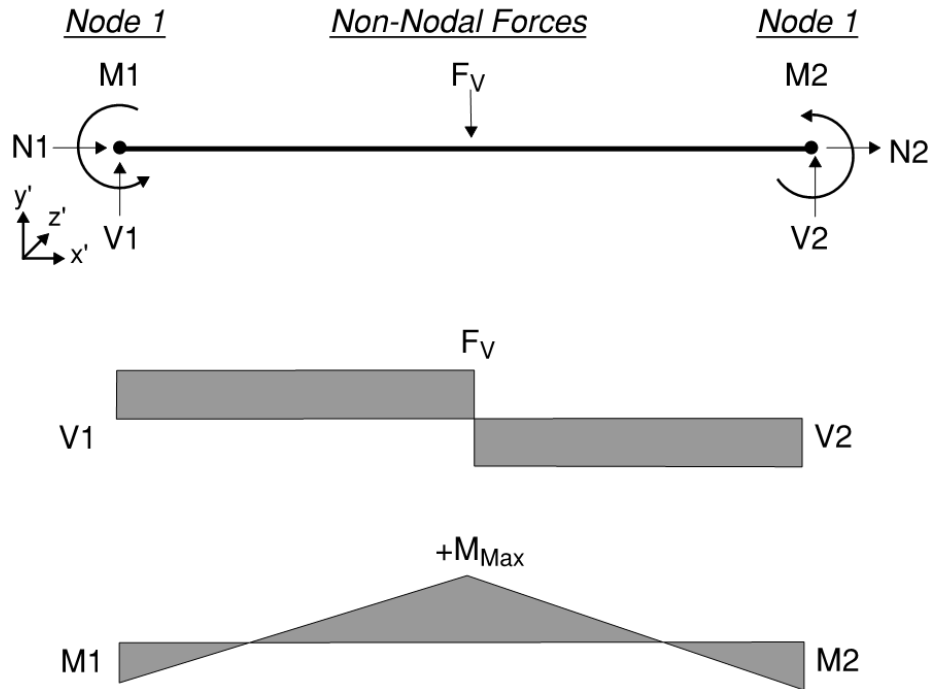


Figure 17: Shear and Bending Moment Due to Applied Loads

5.2.4 Stress and Strain

The strain calculated in the Truss and Frame Modules consists of axial and bending stress and strain. The stress and strain function is included in the StrainStressView file and initiated by the user activation. The Stress and Strain function flow chart is described in Figure 18.

5.2.4.1 Stress and Strain: Process 1.0 and Process 2.0

The Point of Interest, which is a movable designated point along the structure that can reveal the structure's stress and strain at that point, is recorded, and the Member ID and Local Distance are stored in the StructuralView Properties and are used in Process 1.0 to determine the selected member when the Stress and Strain function is activated. In the *iStructure* application, The Point of Interest is represented by a red dot symbol; this point characterizes the structure's stress and strain profile.

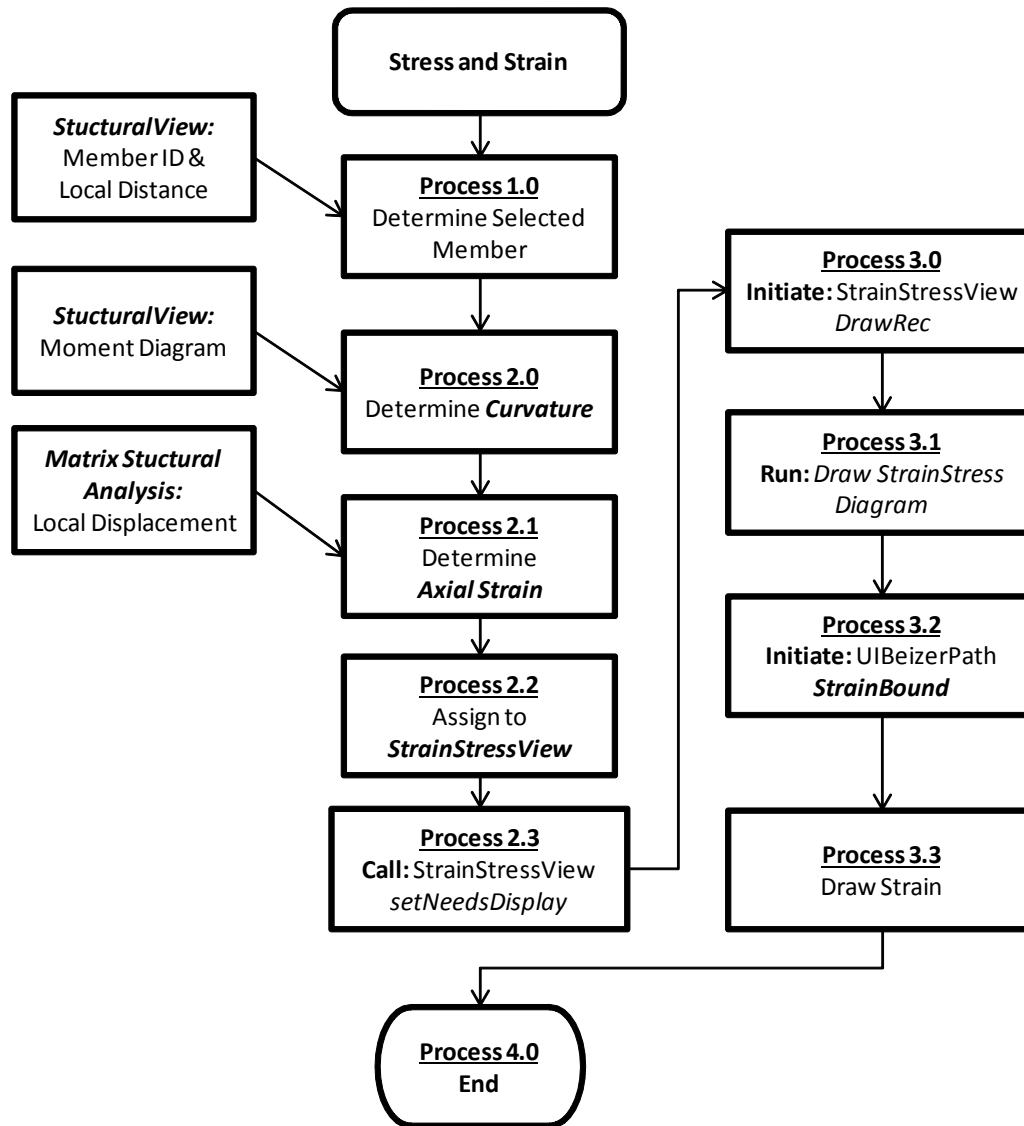


Figure 18: Stress and Strain Flow Chart

In Process 2.0, the selected member's moment diagram and material properties (**E** and **I**) are retrieved to calculate the Beam Curvature (κ) at a local distance (x) along the member, as shown in Eq. (18):

$$\kappa = \frac{M(x)}{EI} \quad (18)$$

In Process 2.1, the local displacement and member length are used to determine the axial strain. In Process 2.2, the Beam Curvature and Axial Strain are assigned to the

StrainStressView's properties and the StrainStressView's *setNeedsDisplay* is called to initiate StrainStressView's *DrawRect* function.

5.2.4.2 Stress and Strain: Process 3.0

Once the StrainStressView's *DrawRect* function is initiated, the *Draw StrainStressView Diagram* function is called. In Process 3.2, UIBezierPath object StrainBound is initiated and the strain profile is drawn according to Eq. (19):

$$\varepsilon(y) = -\kappa y + \varepsilon_{axial} \quad (19)$$

Where y is the vertical distance from the member's neutral axis, and κ and ε_{axial} are constant values obtained in Process 2.0. In Process 3.3, once the StrainBound path is completed, the stress and strain diagram is colored using a custom gradient function and the diagram is drawn on the user interface.

5.3 Summary

The processes for the Frame and Truss modules were reviewed and discussed in this chapter. Both modules implemented the same user-interaction and structural analysis function. At the initiation of each module, the StructureView object and its properties are initialized. The user-interaction functions, such as *Main Run Loop* and *Gravity* function, are called and any alteration of the structure is recorded and stored in the StructureView object for structural analysis. The structural analysis is performed using the *Matrix Structural Analysis* function.

The Matrix Structural Analysis function is used to determine the Truss and Frame structure's characteristics, as described in Chapter 4. The results of the Matrix Structural Analysis Function are then displayed and animated for the user for the respective Frame and Truss modules.

Chapter 6: Alternative Methods for Structural Analysis and Computer Algorithms

6.0 Introduction

An alternative method, employing predetermined formulas and equations in lieu of the Matrix Structural Analysis, as discussed in Chapter 4, is implemented for the Beam module, making computation less demanding and coding routines less complicated. The predetermined conventional beam formulas are library functions that can be imported into the main function when needed. This method is used to determine the beam's characteristics, such as beam deflection and reactions, shear and moment diagrams, and influence line.

6.1 Alternative Methods

6.1.1 Beam Deflection, Shear, and Moment Diagrams

Predetermined beam formulas are incorporated for all of the beam structural types. The beam formulas representing the beam's deformation, shear, and bending moment characteristics are obtained from the AISC construction manual (American Institute of Steel Construction, 2011) based on Euler-Bernoulli's beam method. The principle of superposition is used to account for multiple applied loads.

6.1.2 Continuous Beam

Continuous beams are treated as a single span beam, with internal support reaction acting as applied loads. The reactions of the supports are determined using the three-moment equation.

6.1.3 Influence Line

The influence line illustrates reactions or internal forces experienced at a designated point on a structure while a given load is moving along the structure. As an example, Figure 19 displays the reactions experienced at a support when a unit load is at any given location.

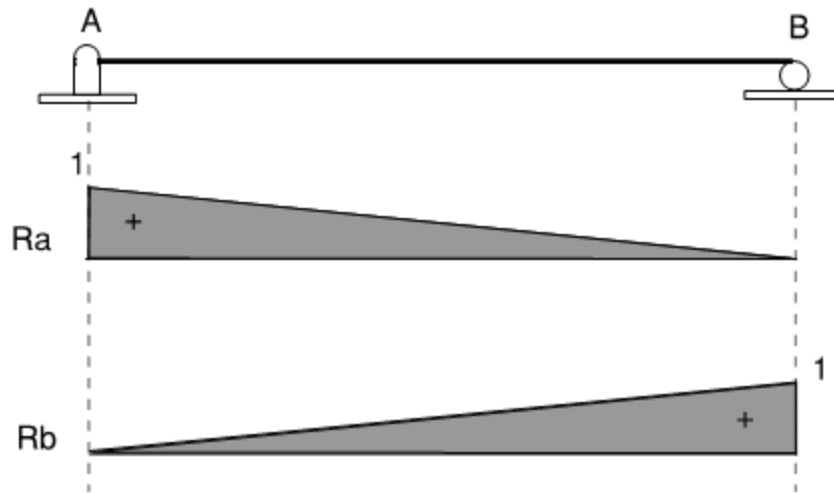


Figure 19: Reaction Influence Line

The influence line can also be used to illustrate shear and bending moments at a designated point in a structure with a unit force moving along the structure, which can determine the critical loading conditions that result in the highest induced internal force for a given structure.

6.2 Computer Algorithms

6.2.1 Beam Module

The Beam module consists of a UIViewController (see Appendix A). The procedures are described in the flow chart in the Figure 20.

6.2.1.1 Beam Module: Process 1.0 and Process 4.0

The processes for the Beam Module are implemented in the same manner as Truss and Frame Module as described Chapter 4. The difference lies in the *Structural Analysis* function, where *Beam Analysis* function is called instead of the *Matrix Structural Analysis*.

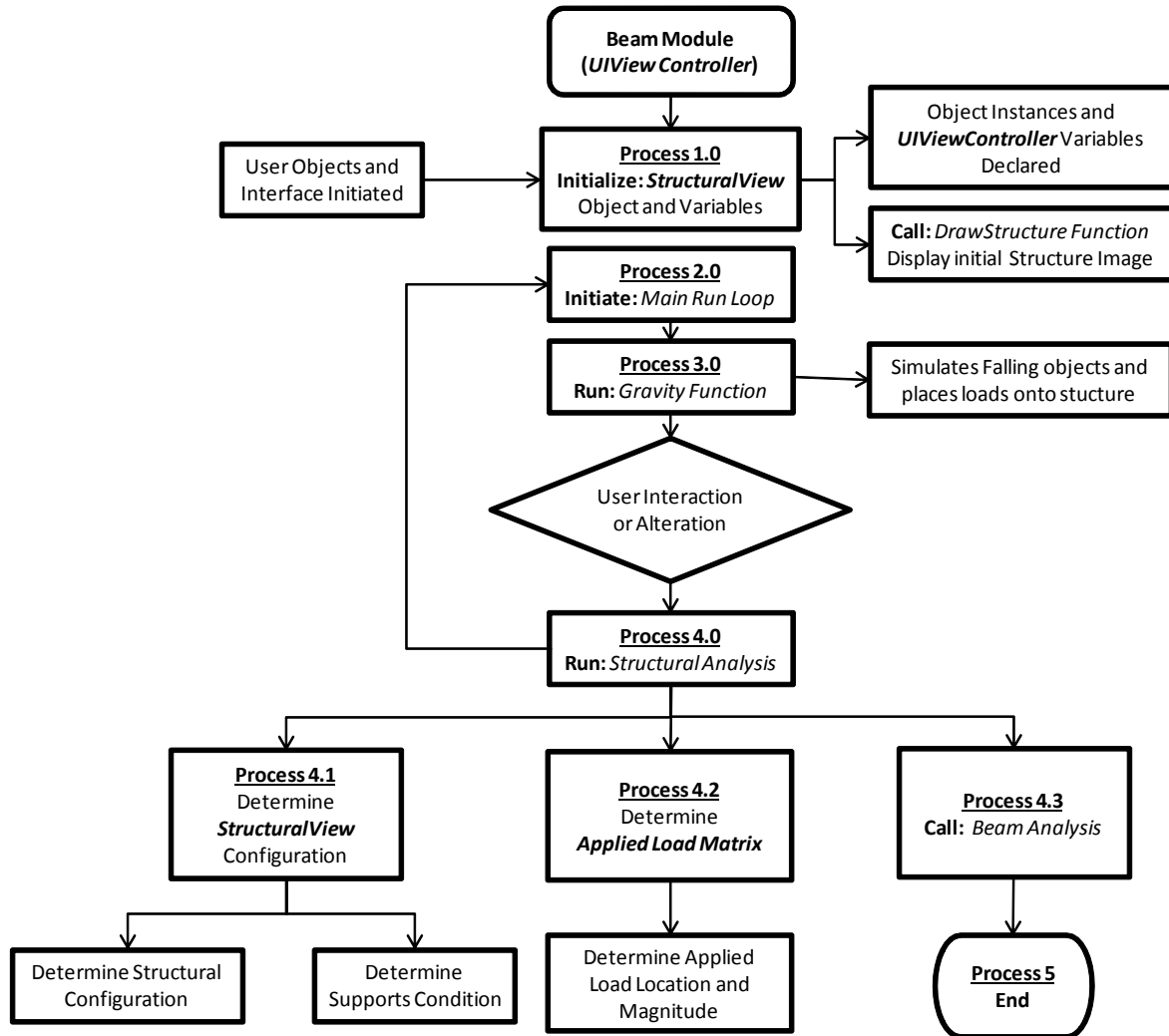


Figure 20: Beam Module Flow Chart

6.2.2 Deflection and Reactions

The Deflection and Reaction procedure and flow chart are described in Figure 21.

6.2.2.1 Beam Analysis: Process 1.0

The Beam Type is determined based on the StructureView's Configuration. Based on the Beam Type, the appropriated beam formula is called. The beam is determined a continuous beam when the SupportMatrix holds more than two Support objects.

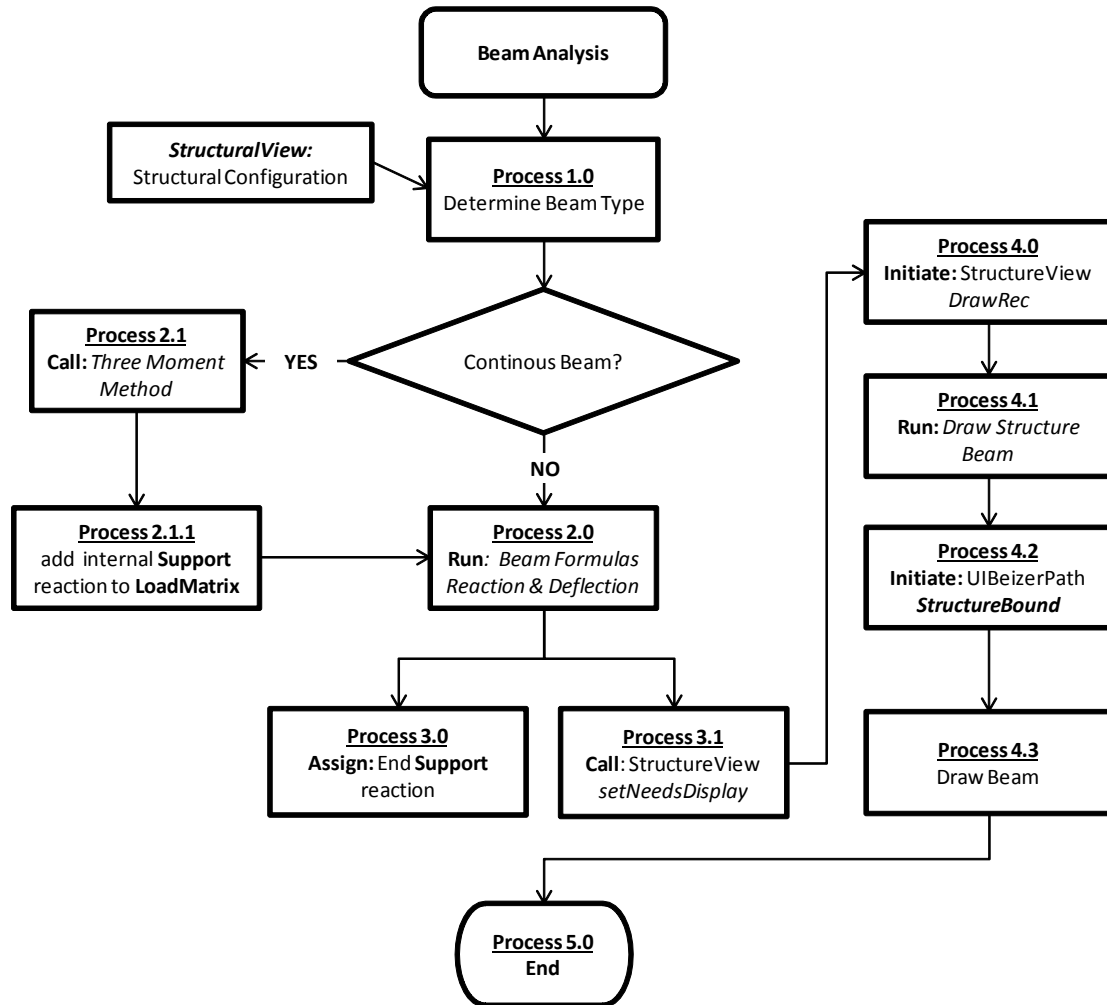


Figure 21: Beam Analysis Flow Chart

6.2.2.2 Beam Analysis: Process 2.0 and Process 3.0

If the SupportMatrix holds more than two Support objects, then Process 2.1 is initiated and the Three Moment Equation is called; the internal support reactions are then determined. In Process 2.1.1, the support reactions are assigned to the Support Object and the internal support reactions are then stored in the AppliedLoadMatrix.

In Process 2.0, the StructureView's Configuration and AppliedLoadMatrix are sent the appropriate beam formula based on the Beam Type to determine the beam deflection and exterior support reactions. The resulting [x, y] deflection points are then assigned to the StructureView's property.

In Process 3.0, the support reactions are assigned to the Support object's UILabel and are displayed in the user interface. In Process 3.1, the StructureView's *setNeedsDisplay* is called to initiate the *DrawRect* function.

6.2.2.3 Beam Analysis: Process 4.0

The *Draw Structure Beam* is called once the *DrawRect* is initiated. In Process 4.2, the UIBezierPath object StructureBound is drawn using the [x, y] deflection points. Once the StructureBound is completed the beam is then drawn on the user interface.

6.2.3 Shear and Bending Moment Diagram

The shear and bending moment diagram procedure and flow chart are described in Figure 22.

6.2.3.1 Shear and Bending Moment: Process 1.0

Once the Beam Analysis is initiated and the structure is drawn, a condition for shear and bending moment diagram is checked.

6.2.3.2 Shear and Bending Moment: Process 2.0 and 4.0

The shear or moment draw function is initiated once the appropriate diagram function is determined. Process 3.0, the *Beam Formula* function for shear or moment is called and iterated through the all the applied loads in the AppliedLoadMatrix and superpositioned into one set of [x, y] diagram points. In Process 4.1, UIBezierPath object DiagramBound is initiated and the diagram points are drawn and displayed on the user interface.

6.2.4 Influence Line

The Influence Line function, described in Section 6.1.3, consists of its own UIViewController file. The Influence Line UIViewController file is structured in the same manner as the Beam Module UIViewController file, but instead of calling the *Beam Analysis* function, the Influence Line function is called. The Influence Line procedure and flowchart is described in Figure 23.

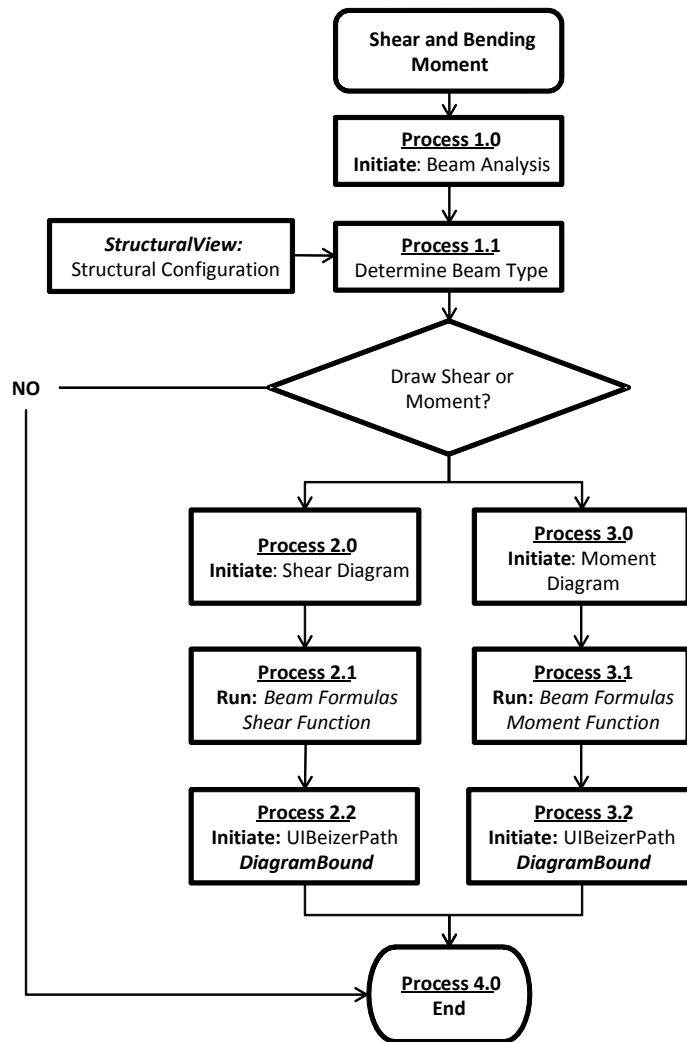


Figure 22: Shear and Bending Moment Flow Chart

6.2.4.1 Influence Line: Process 1.0

Before the Beam Analysis is initiated, a Unit Load object is created and allocated. The Unit Load is given all the properties of a Load object with a unit magnitude and void image property. This is to prevent the Unit Load image from being displayed on the user interface during the analysis. Iteratively, the Unit Load is moved across the beam structure at an incremental distance and then sent to the Beam Analysis in order to record and store the desired influence line value. This process is repeated until the Unit Load object has moved across the beam structure. When the Unit Load is sent to the Beam Analysis, a condition is checked to determine which type of Influence Line needs to be generated.

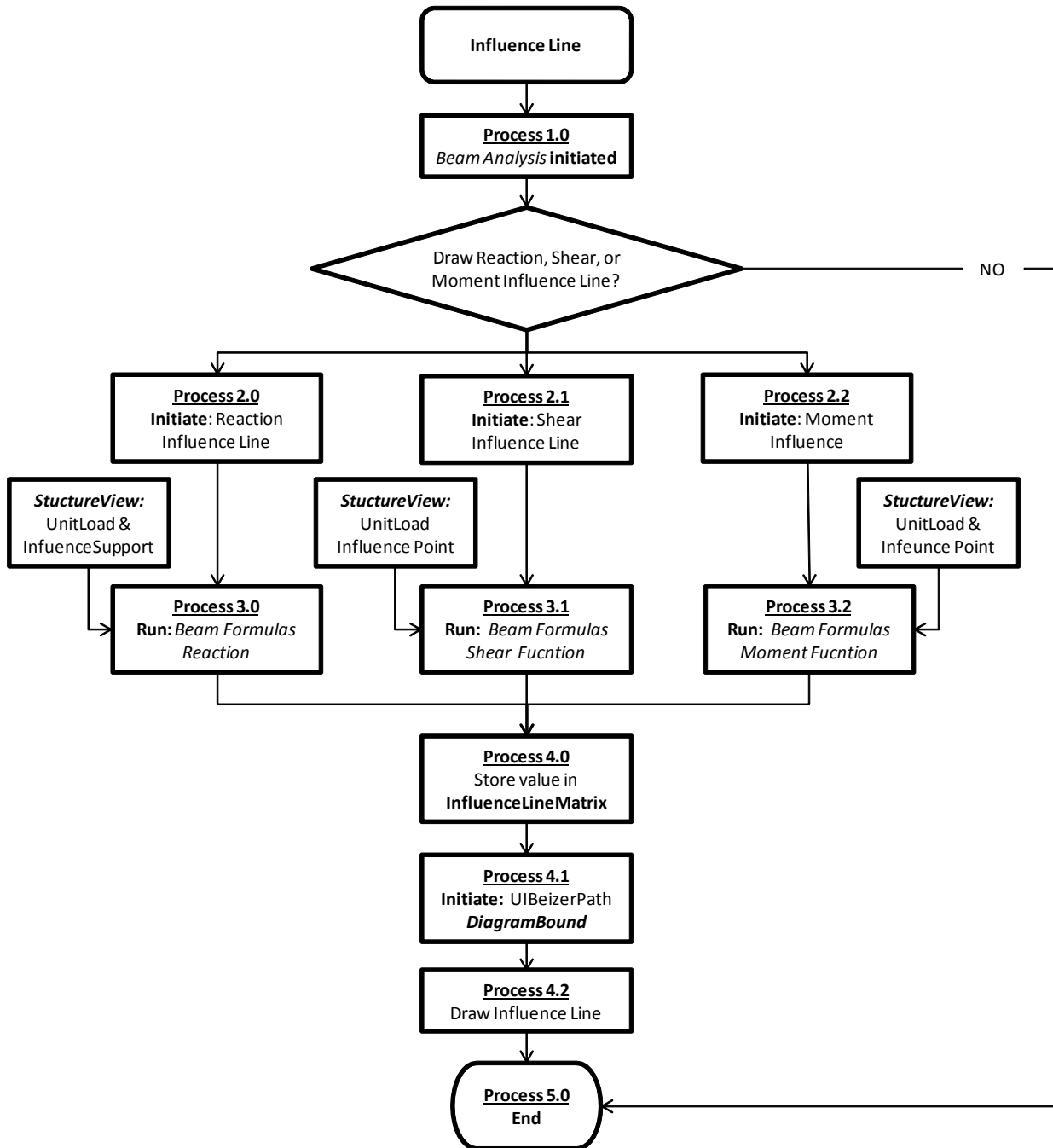


Figure 23: Influence Line Flow Chart

6.2.4.2 Influence Line: Process 2.0 and Process 3.0

Once the Reaction Influence Line function is initiated, the Unit Load properties and selected Support object are sent to the Beam Formulas Reaction function. The reactions of the

selected Support object are then stored in an InfluenceLineMatrix. This process is repeated until the Unit Load has crossed the beam structure.

For the Shear and Moment Influence Line function, the Unit Load properties and a selected point on the beam structure called the Influence Point are sent to either to the *Shear* or Moment Formulas Reaction functions. The shear or moment at the Influence Point with the given Unit Load is then determined and stored in the InfluenceLineMatrix. This process is repeated until the Unit Load has crossed the beam structure.

6.2.4.3 Influence Line: Process 4.0

The values obtained from the Influence Line sub-function are stored in the InfluenceLineMatrix which, at the end of the sub-function process, is stored in the StructureView's property and the StructureView's setNeedsDisplay is called.

Process 4.1, after the StructureView's DrawRect is initiated the InfluenceLineMatrix is sent to the UIBezierPath object DiagramBound to be drawn. The Influence Line is drawn on the user interface once the Diagram Bound path is completed.

6.3 Summary

Alternative methods employed were discussed in this chapter. Predetermined formulas and equations were used to evaluate the beam structure. The beam deflection, reaction, and shear and moment diagram were obtained from the predetermined formulas. Since predetermined formulas are stored in the application's library, less computation demand is required for the structural analysis. This feature allows for multiple structural analysis iterations which were used to develop the beam influence line.

Chapter 7: Application Verification

7.0 Introduction

The purpose of *iStructure* is to provide an enhanced and interactive visual display of structural concepts. For that reason the numerical values of deflection, internal forces, and stress and strain are not displayed in the application. In this section, the performance of *iStructure* is validated using the commercial software SAP2000. Each module and its components are validated conceptually and, in some cases, numerically.

7.1 Beam Structure Module Verification

The Beam Structure module utilizes alternative methods for determining the structural behavior, as discussed in Chapter 6. The resulting structural behavior based on these methods will be compared and validated for structure deformation and reaction, shear and bending moment diagrams, and stress and strain using different structural configurations.

7.1.1 Structure Deformation and Reactions

The beam deformation and reaction are validated through different beam connection cases for a one-span and two-span beam. The loading condition will vary randomly between cases to broaden the confidence of beam deformation and reactions. In Table 4, various loads are applied to a one-span beam for different beam connections. In Table 5, various loads are applied to a two-span beam for different beam connections. Table 5 provides an example of multiple stacked loads, which the *iStructure* application is programmed to treat as applied loads.

Table 4: Beam Deflection and Reaction with SAP2000 Results

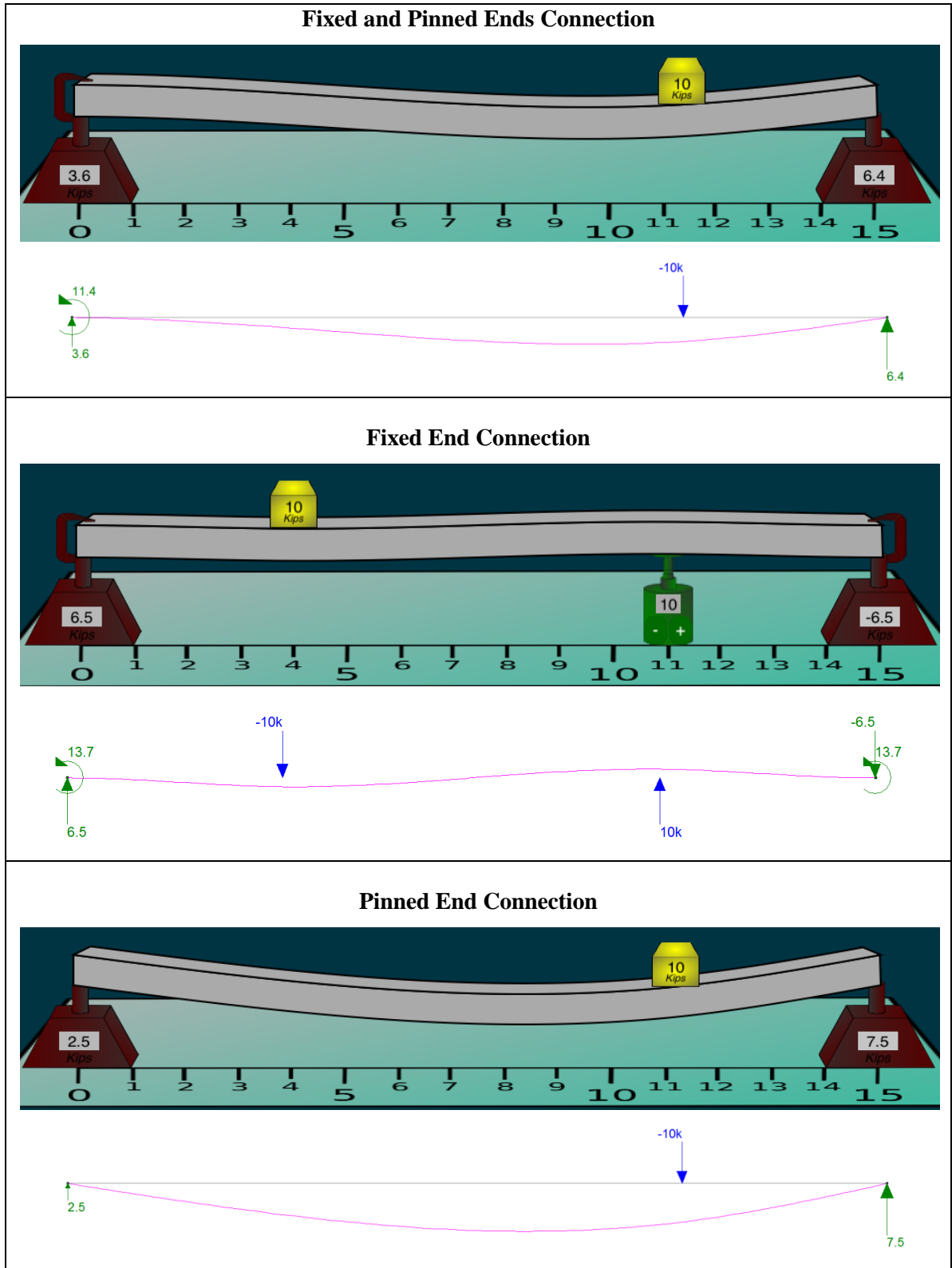
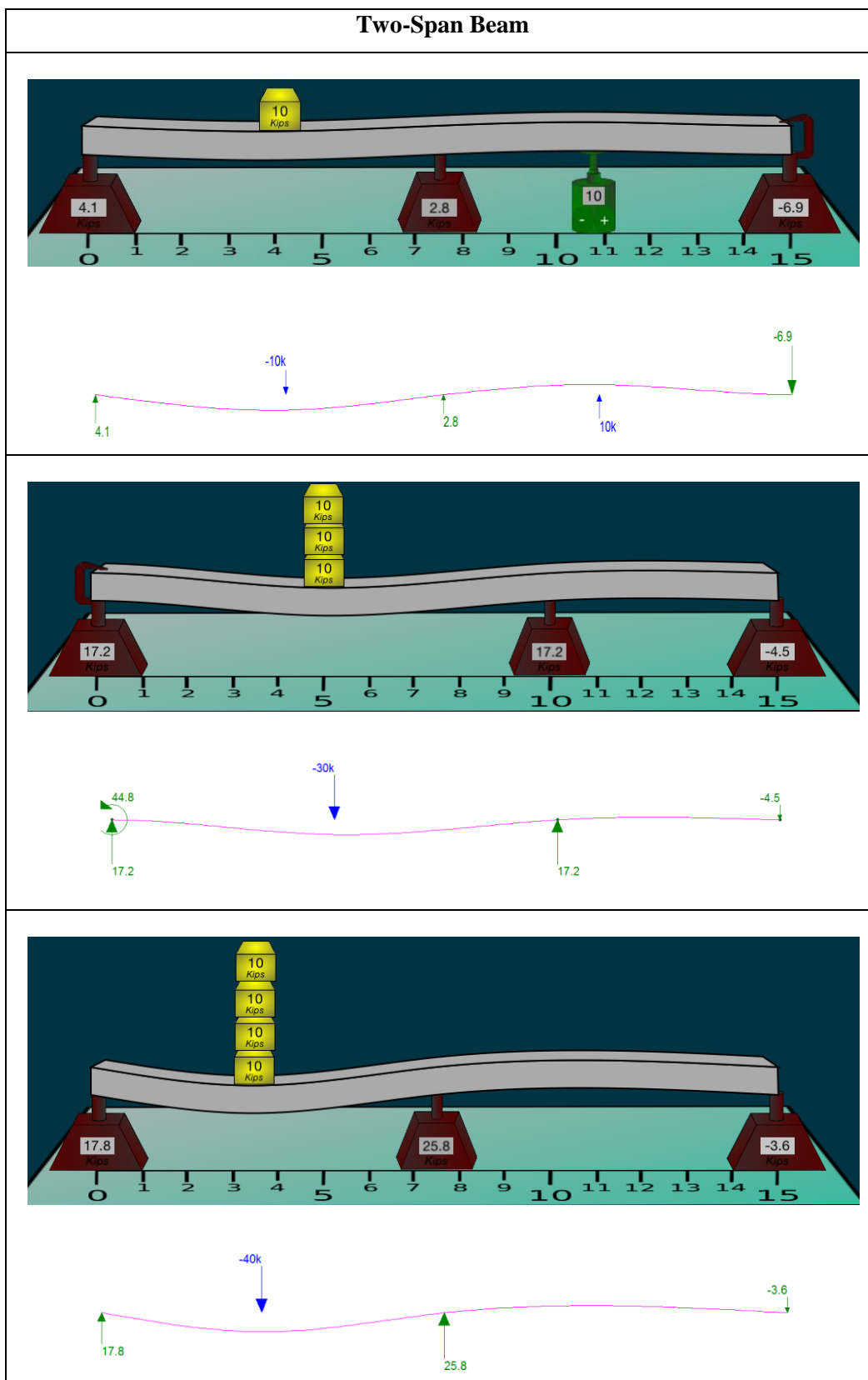


Table 5: Beam Deflection and Reaction with SAP2000 Results for Two-Span Beam



7.1.2 Shear and Bending Moment Diagram

The shear and bending moment validations for the beam structure are developed in the same manner as the beam deformation and reactions, and are shown in Table 6 through Table 9. However, Risa-2D and the *iStructure* applications employ different sign conventions for the moment diagram, which results in an inverse comparison.

Table 6: Beam Shear and Moment Diagram with SAP2000 Results

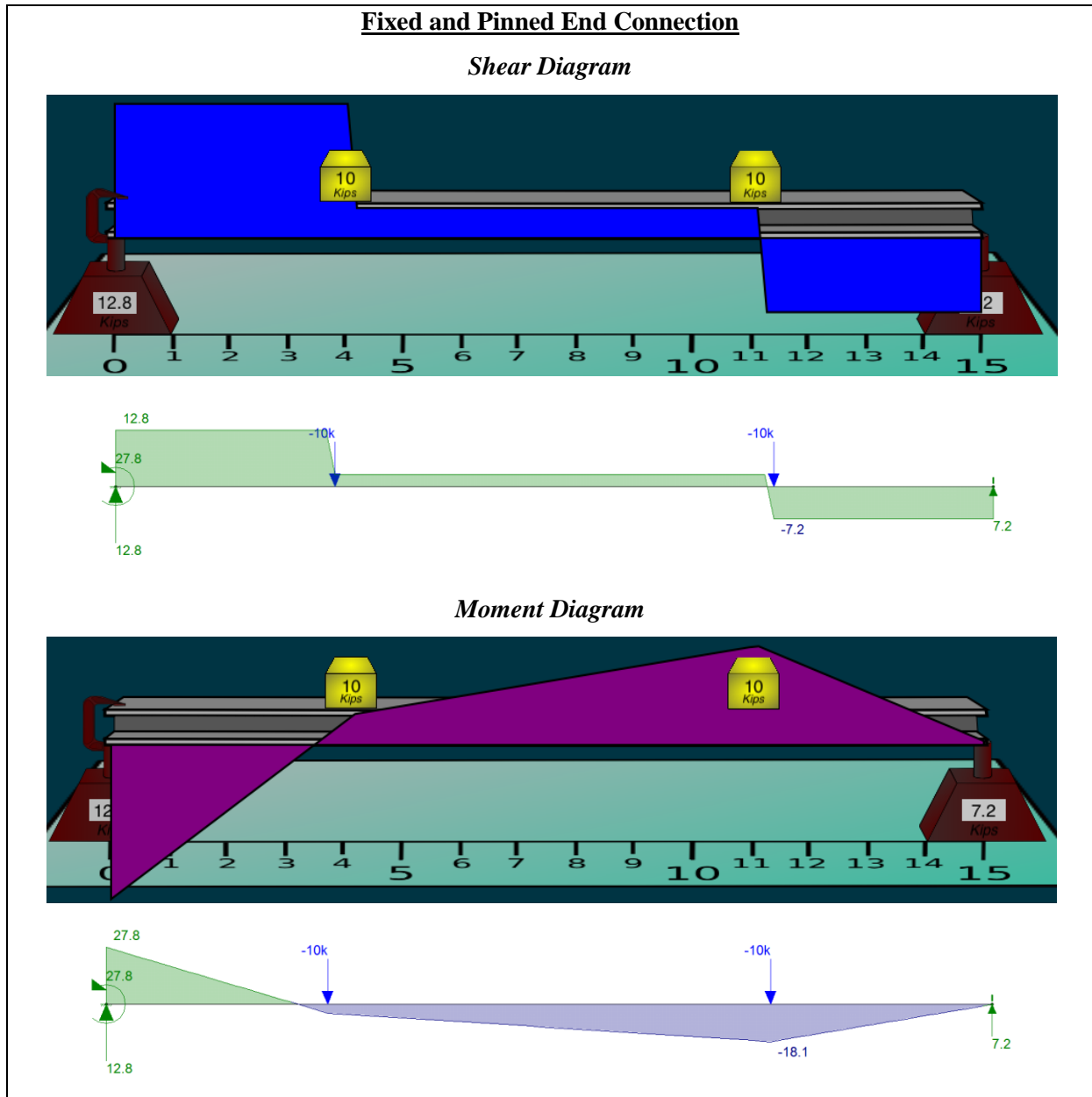


Table 7: Beam Shear and Moment Diagram with SAP2000 Results for Fixed Connections

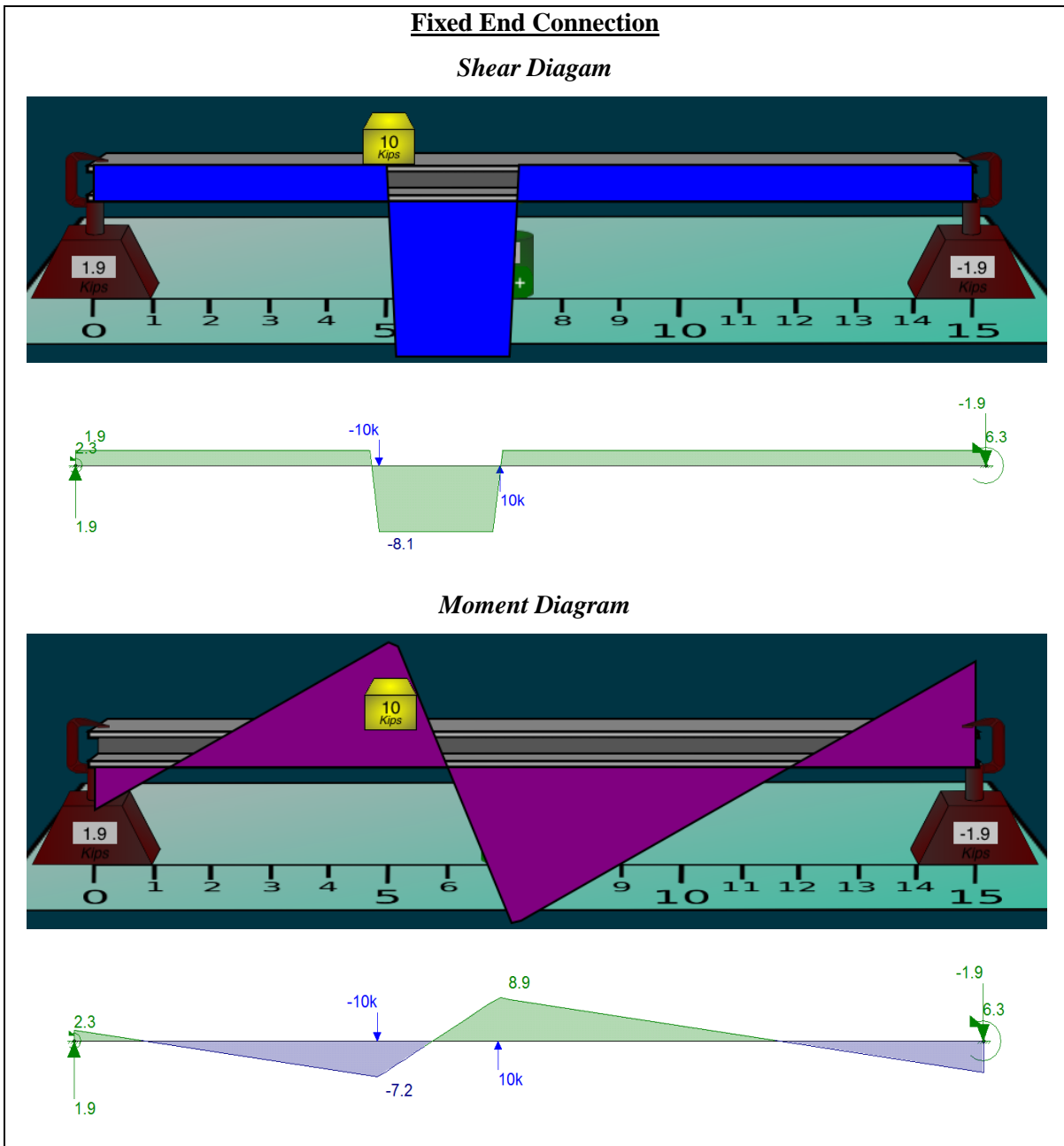


Table 8: Beam Shear and Moment Diagram with SAP2000 Results

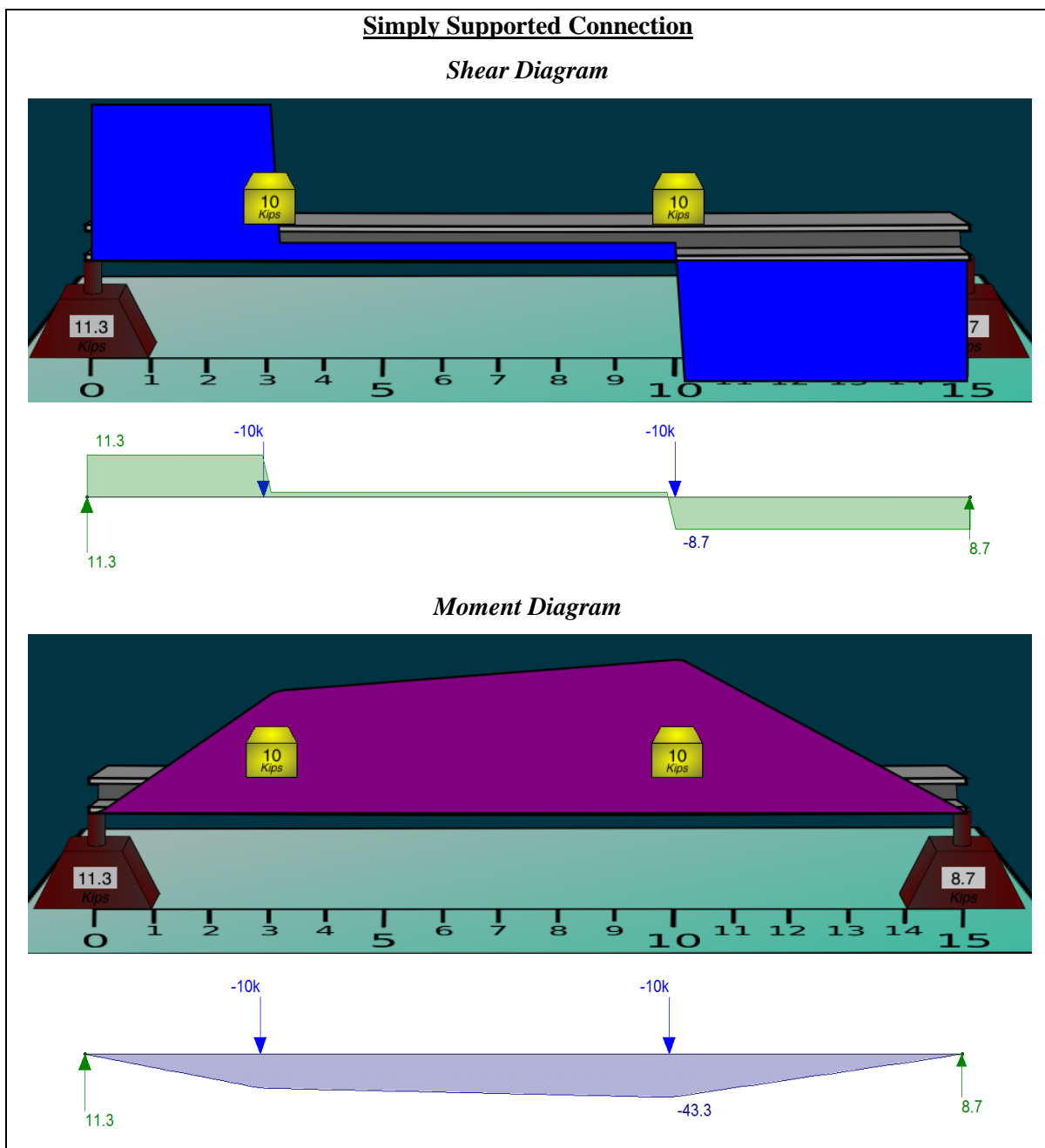
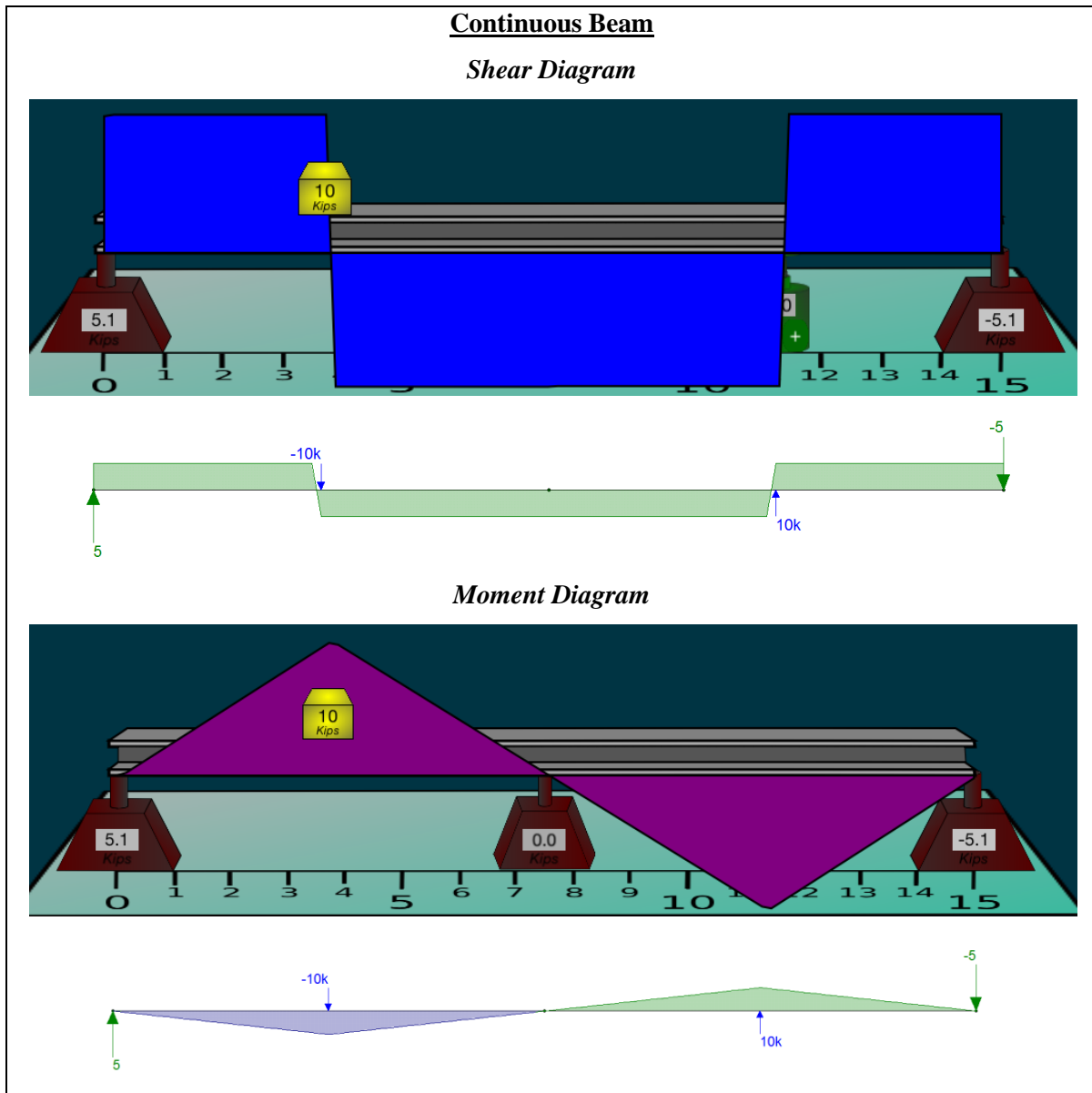


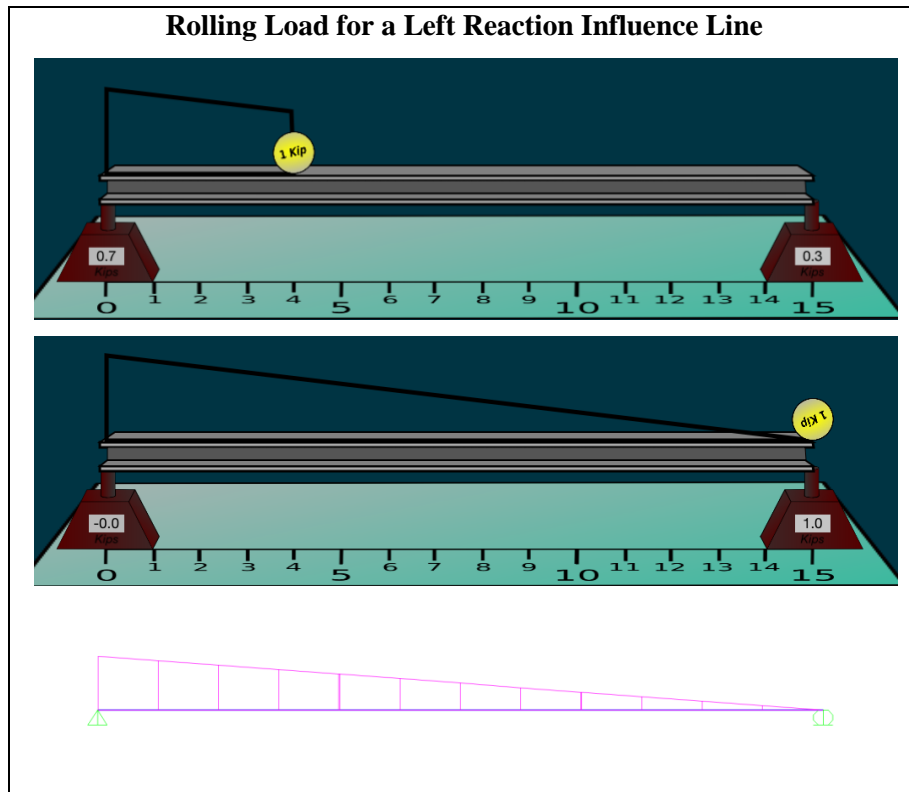
Table 9: Beam Shear and Moment Diagram with SAP2000 Results for Continuous Beam



7.1.3 Beam Influence Line

As part of the *iStructure* application, the beam influence function has an animation feature that simulates a rolling load across the beam structure while revealing the influence line, shown in Table 10.

Table 10: Beam Influence Line Animation with SAP2000 Results



In *iStructure*, the reaction influence line is drawn for a selected support reaction chosen by the user. For the shear and moment influence line, a red dot symbol along the beam represents the point of interest. The influence line for support reaction, shear, and moment are validated using SAP2000's Moving Load function, as can be seen in Table 10 through Table 13. However, SAP2000 and *iStructure* application employ different sign conventions for the shear influence line, resulting in an inverse comparison. These Influence Line validations are shown in the following tables.

Table 11: Influence Line for Support Reaction with SAP2000 Results

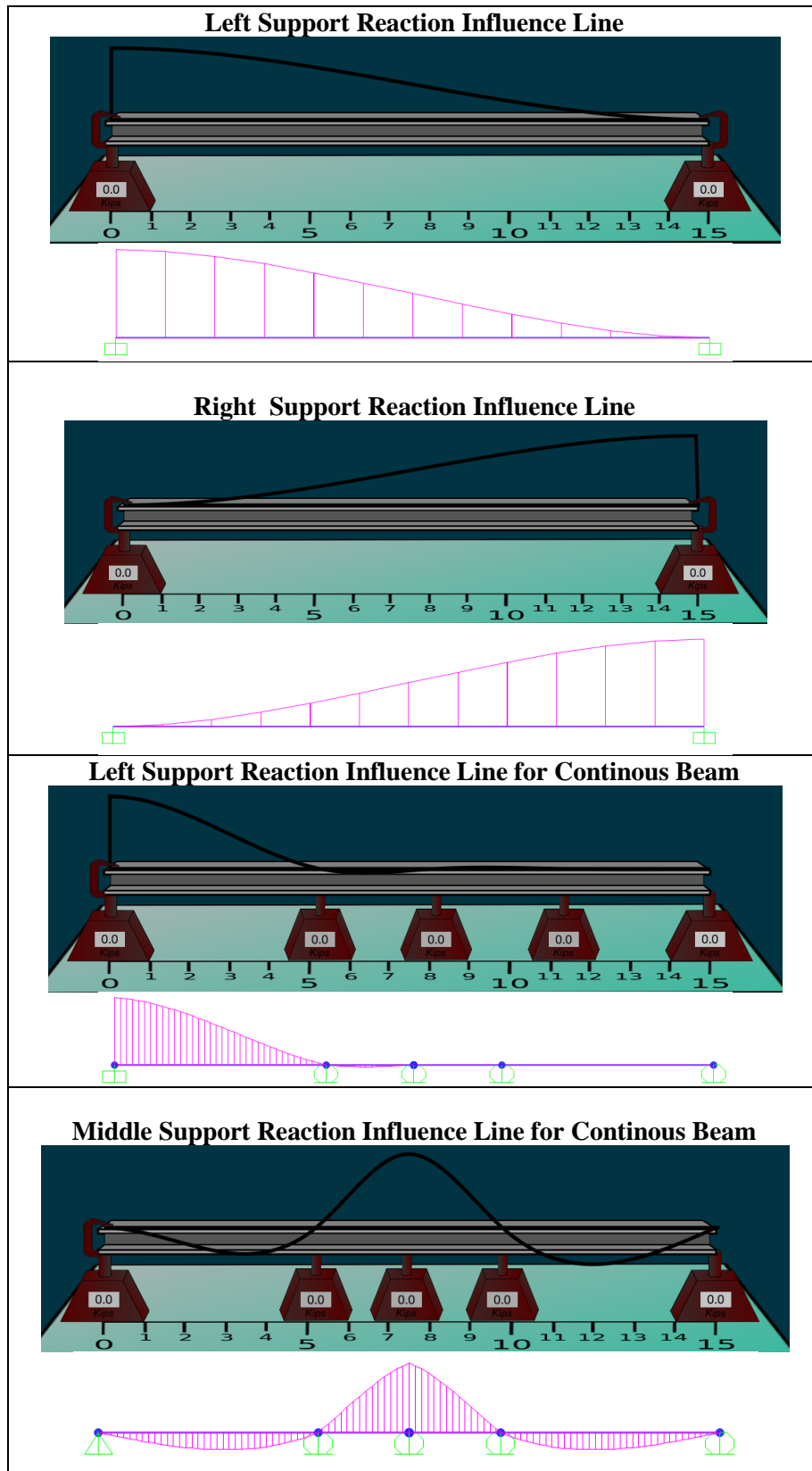


Table 12: Influence Line for Shear Force with SAP2000 Results

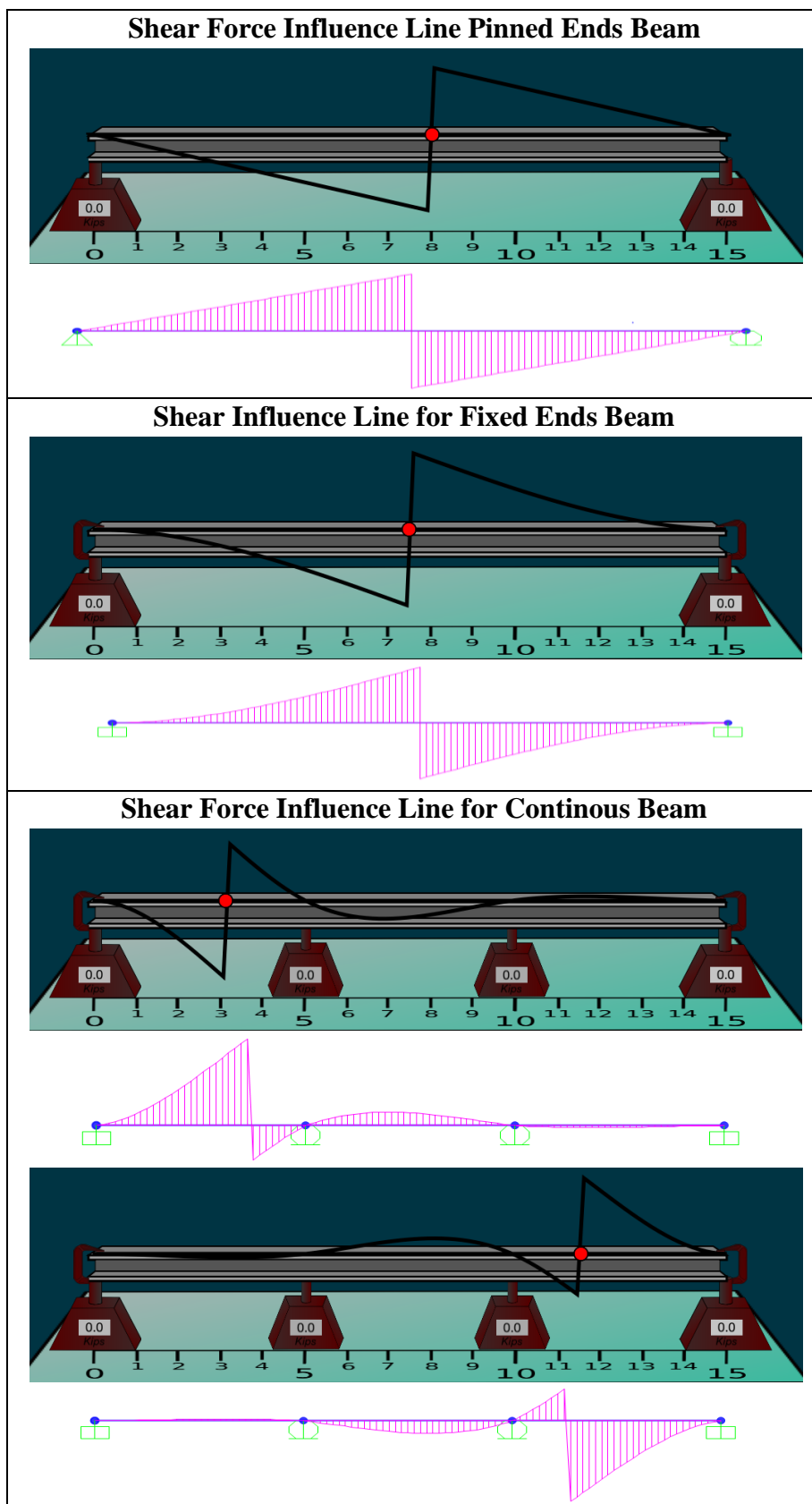
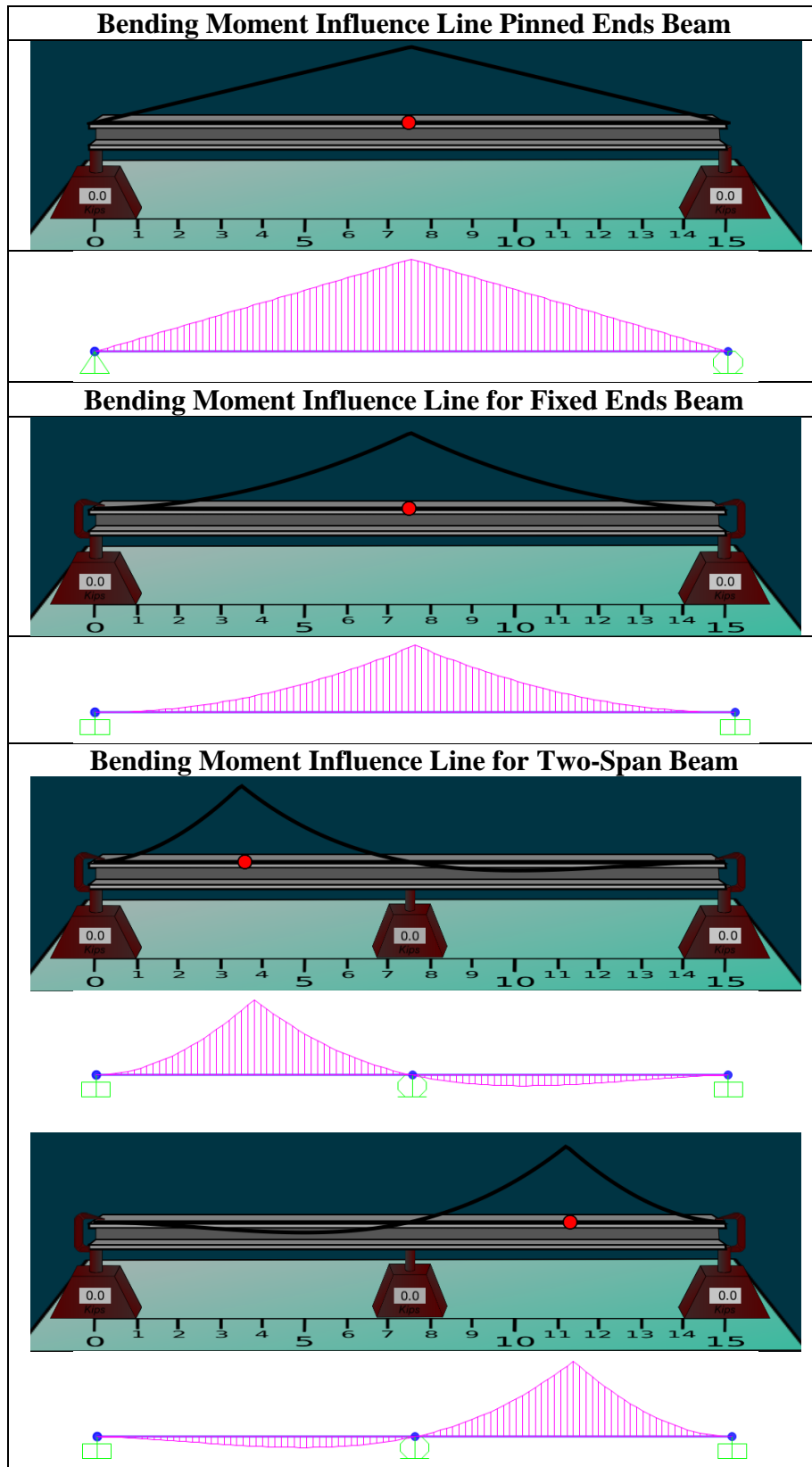


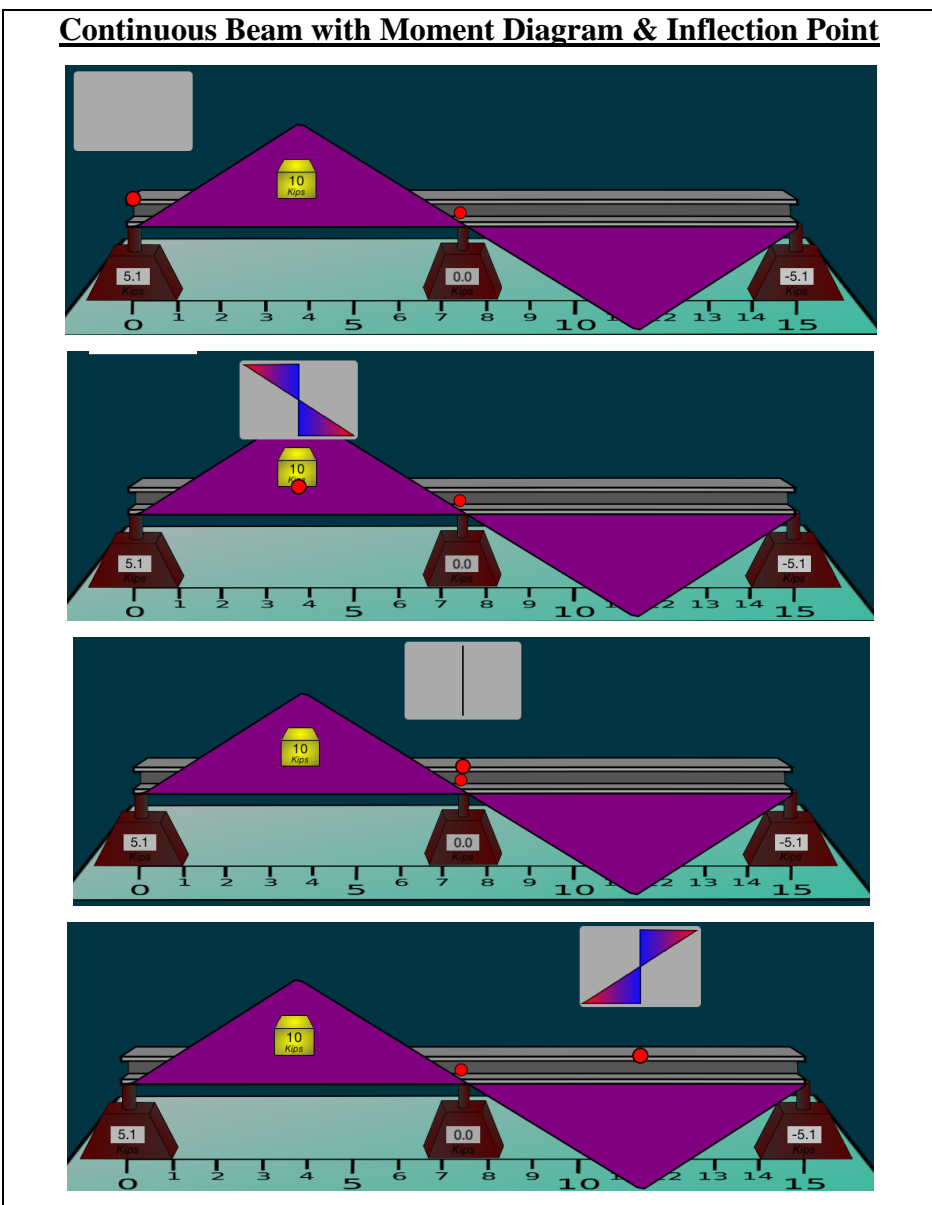
Table 13: Influence Line for Bending Moment with SAP2000 Results



7.1.4 Stress and Strain Diagram

The stress and strain diagrams are directly developed from the bending moment, as described in Section 5.2.4. As a result, the validation of the stress and strain is performed through conceptual and visual validation. Table 14 illustrates a two span beam with a downward force at the mid-span of the left span and an upward force at the mid-span of the right span. As can be seen in Table 14, the stress and strain diagram matches the behavior of the moment diagram.

Table 14: Beam Stress and Strain Diagram with SAP2000 Results



7.2 Truss Module Verification

The Truss Module is validated through verification of the module's components. The Truss Module employs the matrix structural analysis for a 4-DOF element structure. The structure's deflection, reaction, axial diagram, and strain diagram will be validated.

7.2.1 Structure Deformation and Reaction

In the *iStructure* application, the Truss Structure does not have multiple boundary conditions that will need to be validated, thus the Truss Structure can be validated with fewer cases. Risa-2D is used to validate the Truss Structure deformation and reaction. The structure's deflection and reaction validations are shown in Table 15.

7.2.2 Axial Force Diagram

Since the loading conditions are only applied at the structure's joints, the typical assumptions for truss analysis result in truss members carrying only axial forces. For axial force diagram, orange color regions represent compression forces and blue color regions represent tension force. Risa-2D is used to validate the Truss Structure axial force diagram. The axial force diagram validations are shown in Table 16.

Table 15: Truss Deflection and Reaction with SAP2000 Results

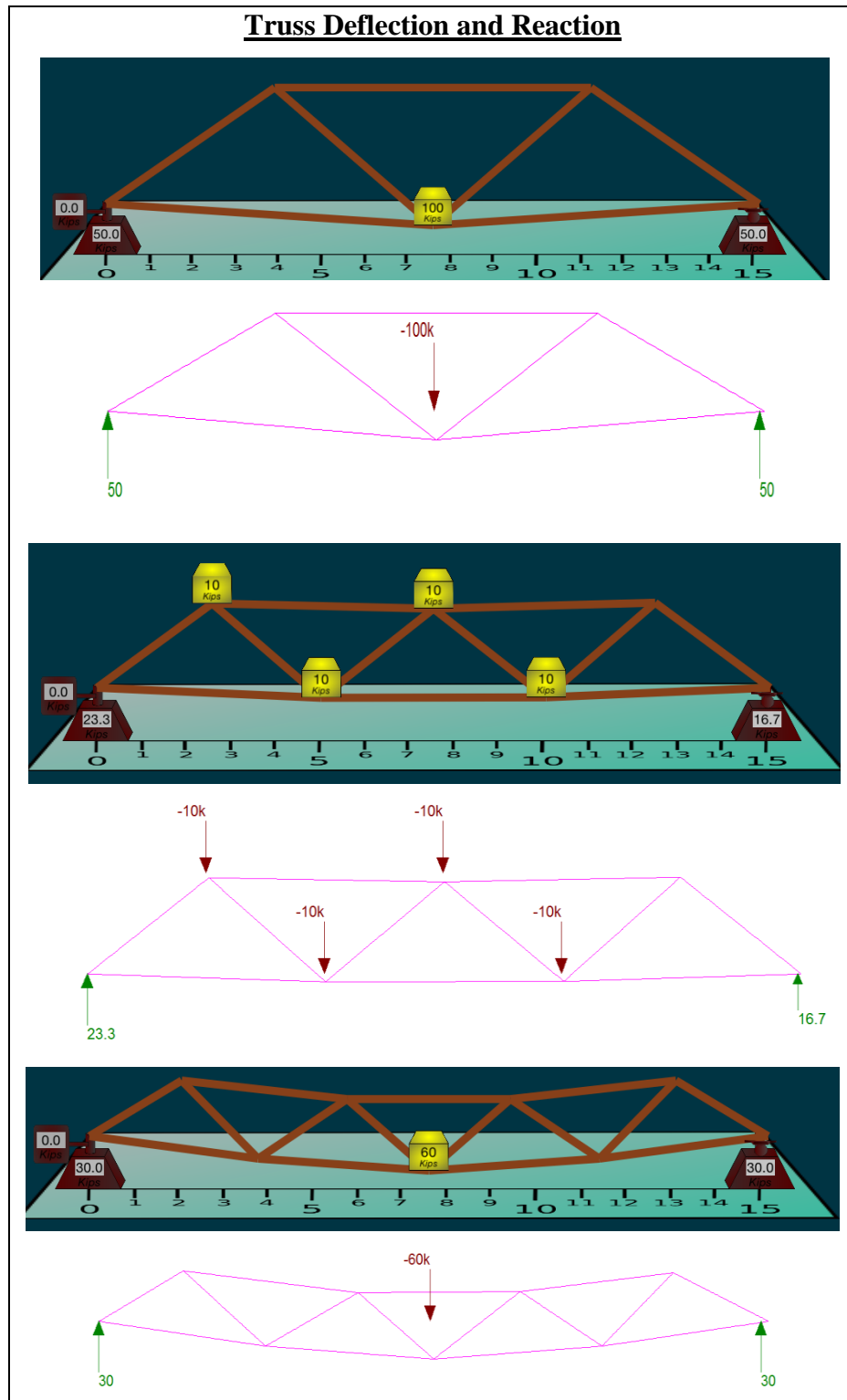
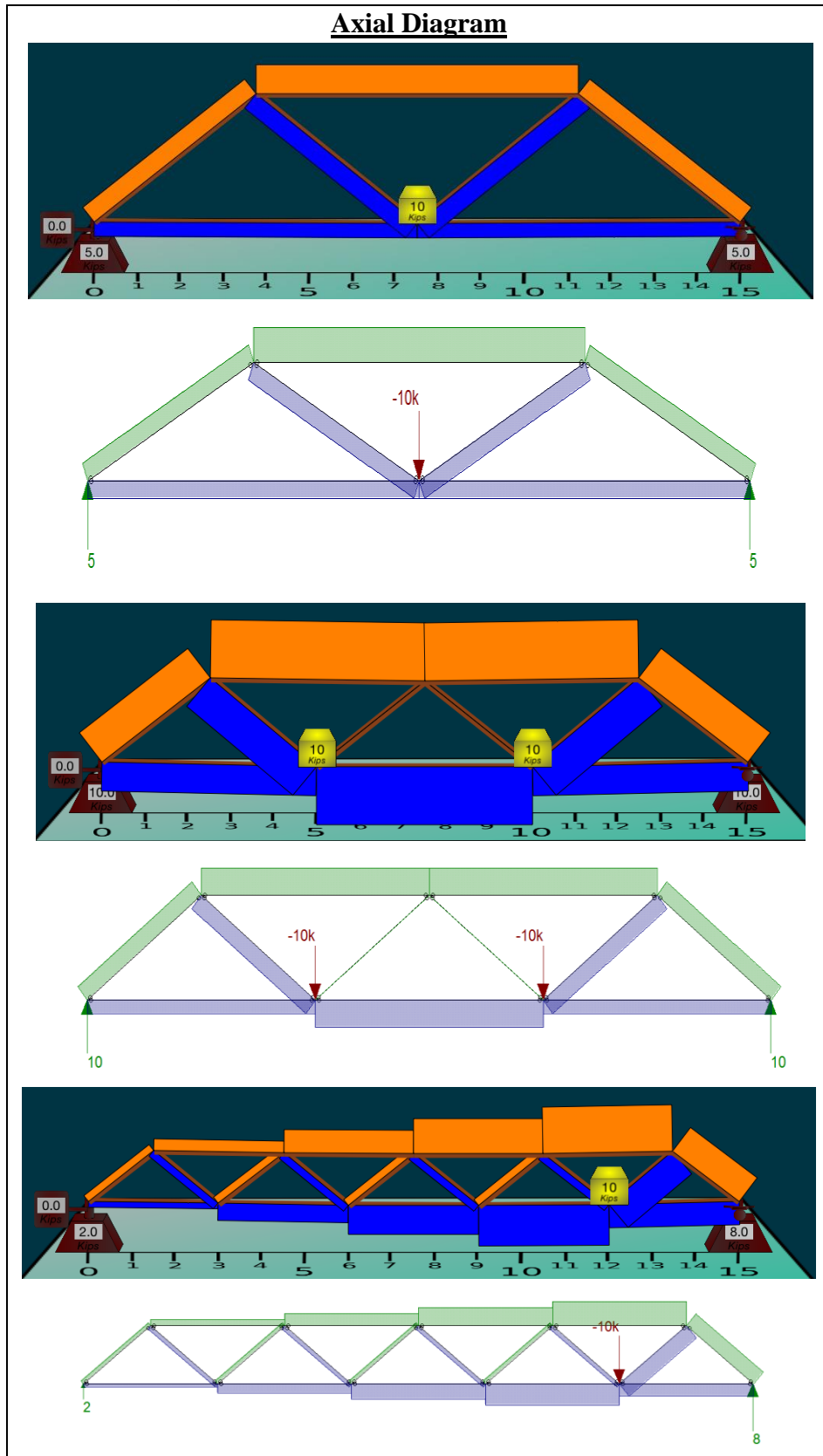


Table 16: Truss Axial Diagram with SAP2000 Results



7.2.3 Stress and Strain Diagram

For the Truss Structure, the stress and strain diagram is only comprised of axial stress and strain as explained in Section 7.2.2. In the same manner of the axial force diagram, the orange color region represents compression (negative) strain and the blue region represents tension (positive) strain. The Truss Structure stress and strain validations are shown Table 17 and Table 18. As you can see in Table 17, the stress and strain matches the behavior of the axial diagram.

Table 17: Truss Stress and Strain Diagram Validation with Axial Diagram

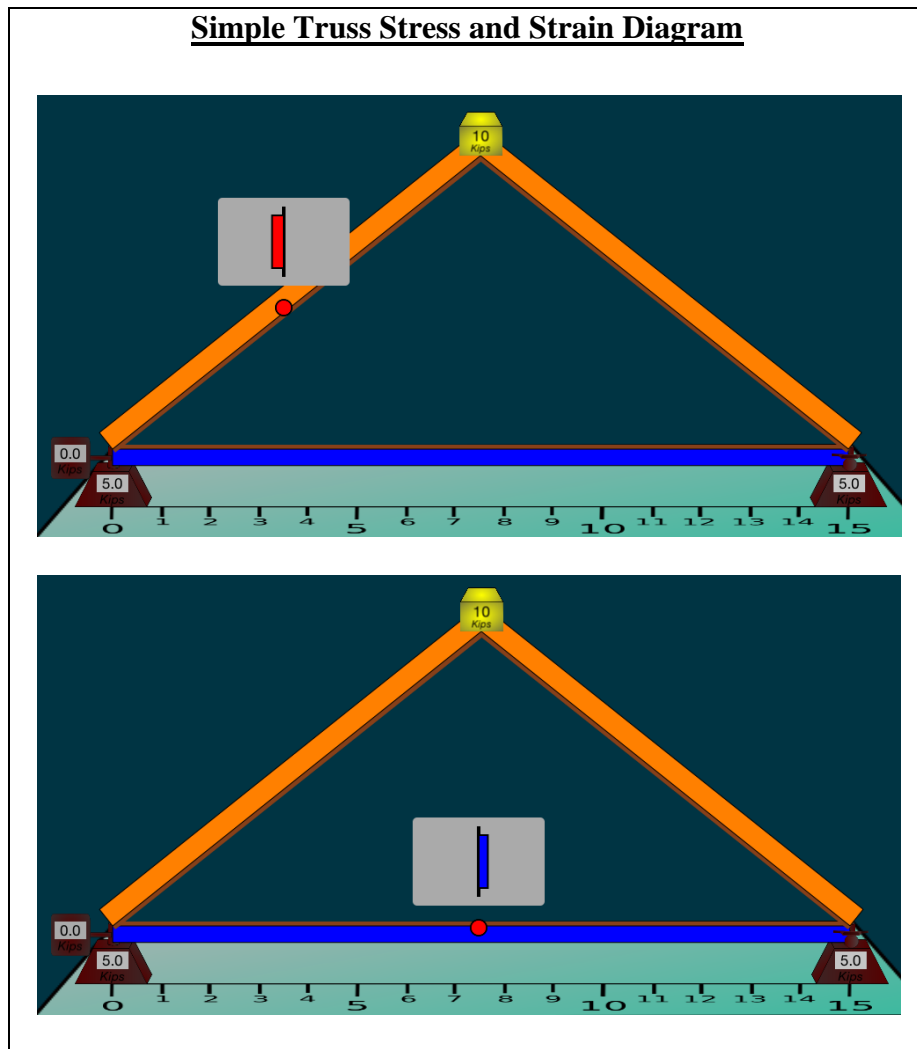
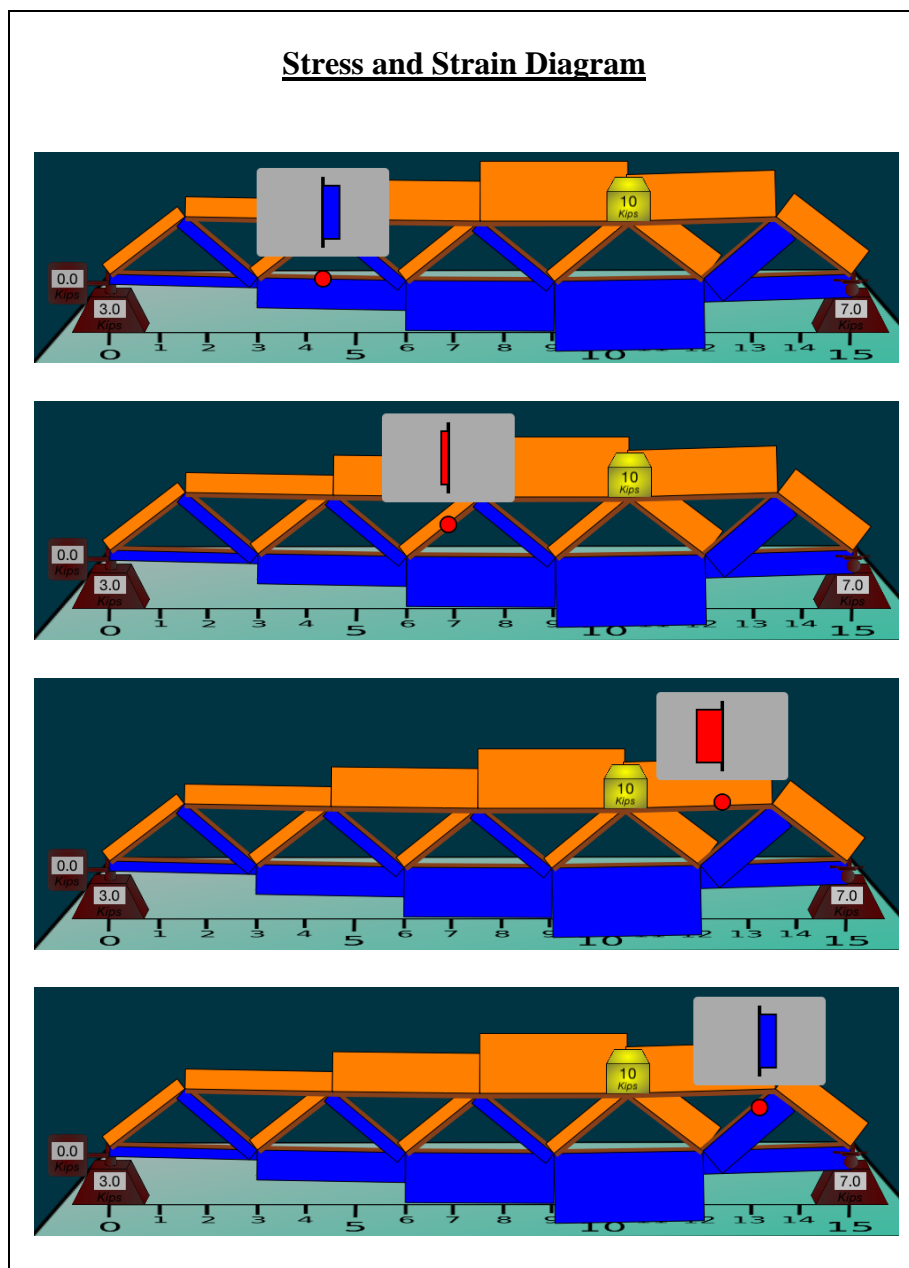


Table 18: Truss Stress and Strain Diagram with SAP2000 Results



7.3 Frame Structure Module Verification

The Frame Structure module will be validated through the module's components. The Frame Structure module employs the matrix structural analysis method for a 6-DOF element structure. This module's deformation and reaction, internal force diagrams, and stress and strain will be validated using Risa-2D through different structural configuration.

7.3.1 Structure Deformation and Reaction

The Frame Structure can consist of multiple configurations involving different support connections, multiple numbers of bays and stories, and varying loading conditions. For simplicity and better visual representation, most of the structure configurations will consist of a single bay and story frame. The Frame Structure's deflection and reaction validations are shown in Table 19.

7.3.2 Axial, Shear, and Bending Moment Diagram

The results of the matrix structural analysis post-processing are used to develop the different internal diagrams. In order to achieve confidence in the diagrams, each internal force type diagram was validated. To increase the confidence in the internal diagrams, both gravity and lateral loads were considered in the validation process. The frame axial force diagrams are validated in Table 20 and Table 21. The frame shear force diagrams are validated in Table 22 and Table 23. Lastly, the frame bending moment diagrams are validated in Table 24 and Table 25.

Table 19: Frame Deflection and Reaction with SAP2000 Results

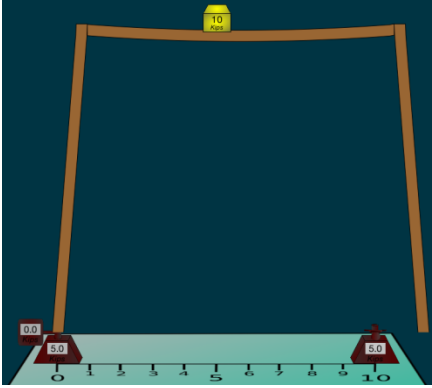
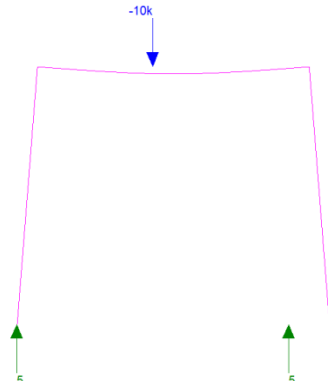
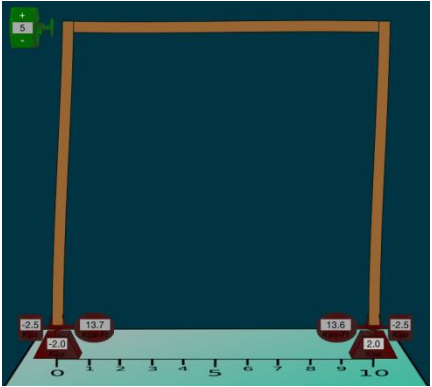
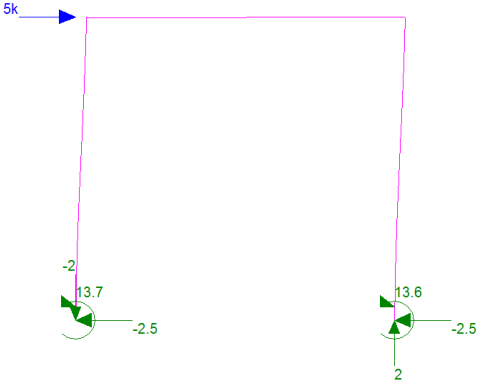
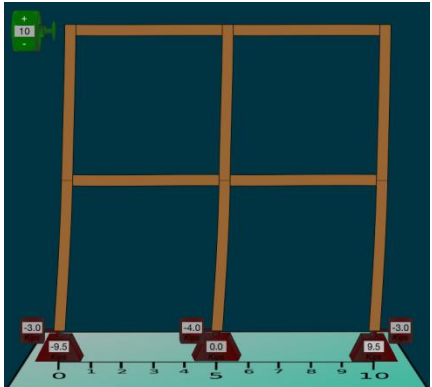
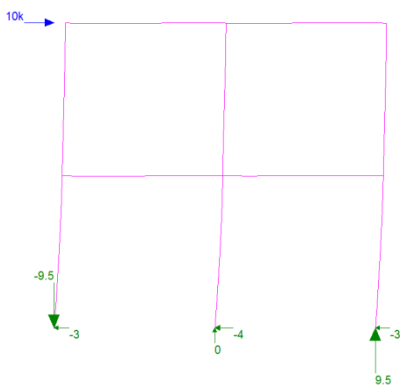
<i>iStructure</i> Frame Deflection	Point Load Center
	
Pinned and Roller End Frame	
	
Fully Fixed End Frame	
	
Fully Pinned End Frame	

Table 20: Frame Axial Diagram with SAP2000 Results for Center Point Loads

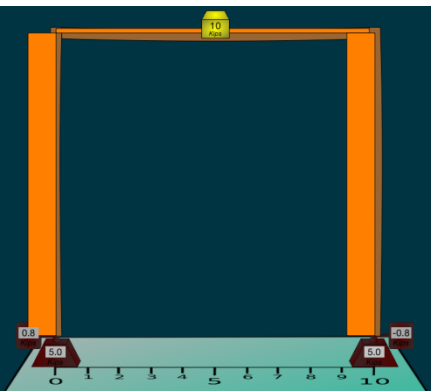
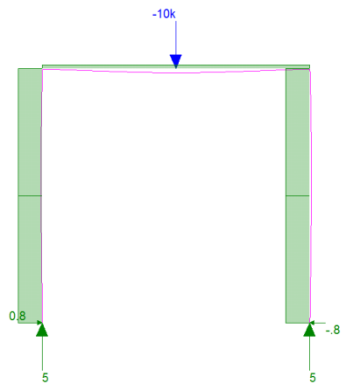
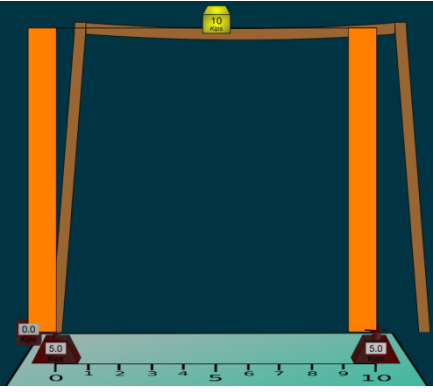
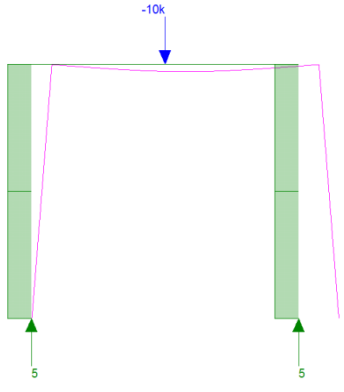
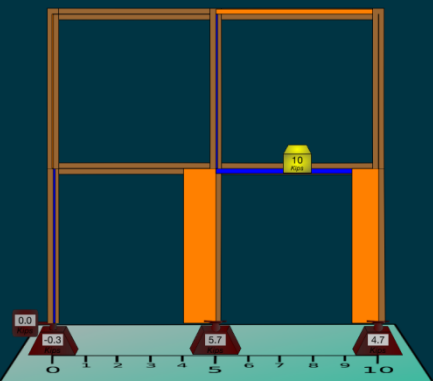
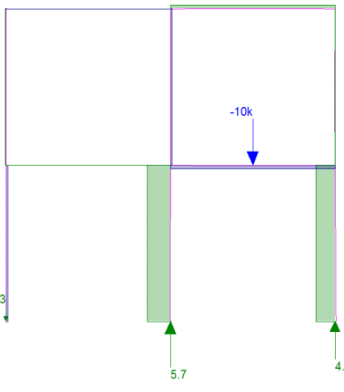
<i>iStructure</i> Axial Diagram	Point Load Center
 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. The beam is highlighted in orange. A yellow callout box above the beam indicates a value of 10. At the base of each column, there are red triangular supports. The left support is labeled with 0.8 and 5.0. The right support is labeled with 0.8 and 5.0. A horizontal axis at the bottom is labeled from 0 to 10.</p>	 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. A blue arrow labeled -10k points down at the center of the beam. The vertical columns are shaded green. At the base of each column, there are green upward-pointing arrows labeled 5. The left column has a value of 0.8 at its base. The right column has a value of -0.8 at its base.</p>
 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. The beam is highlighted in orange. A yellow callout box above the beam indicates a value of 10. The left support is a red triangular pin support labeled with 0.0 and 5.0. The right support is a red triangular roller support labeled with 5.0. A horizontal axis at the bottom is labeled from 0 to 10.</p>	 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. A blue arrow labeled -10k points down at the center of the beam. The vertical columns are shaded green. At the base of each column, there are green upward-pointing arrows labeled 5. The left column has a value of 0.0 at its base. The right column has a value of 5.0 at its base.</p>
 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. The beam is highlighted in orange. A yellow callout box above the beam indicates a value of 10. At the base of each column, there are red triangular supports. The left support is labeled with 0.0 and -0.3. The middle support is labeled with 5.7. The right support is labeled with 4.7. A horizontal axis at the bottom is labeled from 0 to 10.</p>	 <p>The diagram shows a rectangular frame with a horizontal beam and two vertical columns. A blue arrow labeled -10k points down at the center of the beam. The vertical columns are shaded green. At the base of each column, there are green upward-pointing arrows labeled 5.7 and 4.7. The left column has a value of -0.3 at its base.</p>

Table 21: Frame Axial Diagram with SAP2000 Results for Lateral Loads

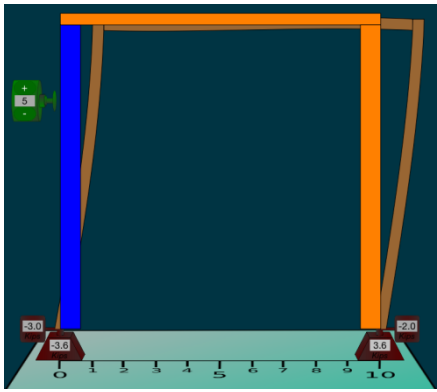
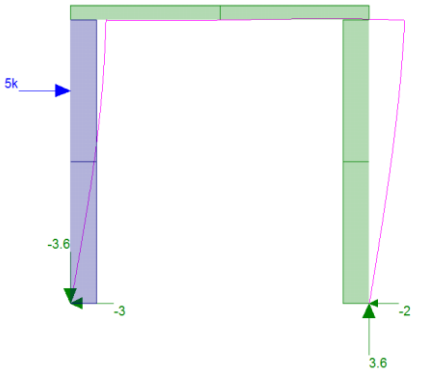
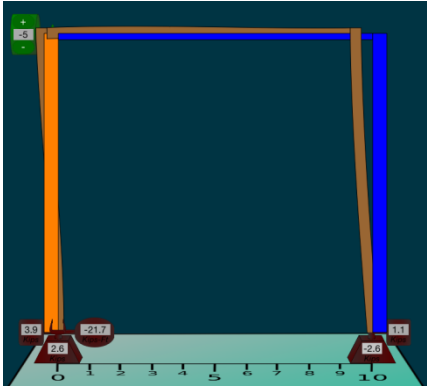
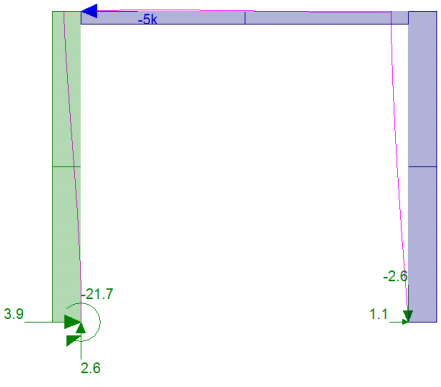
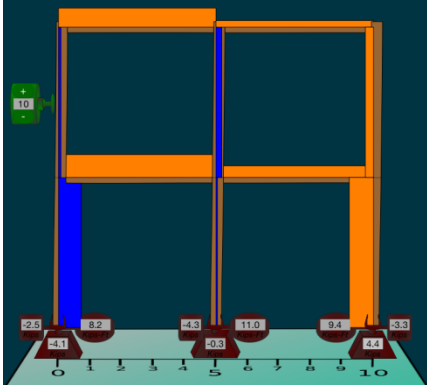
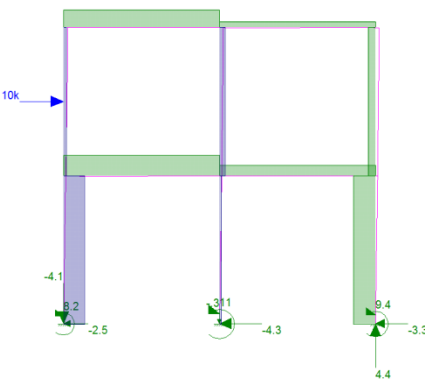
<i>iStructure</i> Axial Diagram	Lateral Load
 <p>The diagram shows a frame with a horizontal load of 5k applied to the left at the top-left corner. The axial force distribution is shown in orange. At the bottom-left support, the axial force is -3.0. At the bottom-right support, the axial force is -2.0. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a horizontal load of 5k applied to the left at the top-left corner. The axial force distribution is shown in green. At the bottom-left support, the axial force is -3.6. At the bottom-right support, the axial force is -2.0. The horizontal axis is labeled from 0 to 10.</p>
Fully Pinned End Frame	
 <p>The diagram shows a frame with a horizontal load of -5k applied to the left at the top-left corner. The axial force distribution is shown in orange. At the bottom-left support, the axial force is 3.9. At the bottom-right support, the axial force is 1.1. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a horizontal load of -5k applied to the left at the top-left corner. The axial force distribution is shown in green. At the bottom-left support, the axial force is 3.9. At the bottom-right support, the axial force is 1.1. The horizontal axis is labeled from 0 to 10.</p>
Fixed and Pinned End Frame	
 <p>The diagram shows a frame with a horizontal load of 10k applied to the left at the top-left corner. The axial force distribution is shown in orange. At the bottom-left support, the axial force is -2.5. At the bottom-right support, the axial force is 3.3. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a horizontal load of 10k applied to the left at the top-left corner. The axial force distribution is shown in green. At the bottom-left support, the axial force is -4.1. At the bottom-right support, the axial force is 9.4. The horizontal axis is labeled from 0 to 10.</p>
Fully Fixed End Frame	

Table 22: Frame Shear Diagram with SAP2000 Results for Center Point Loads

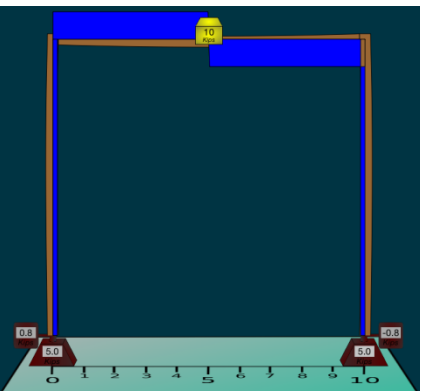
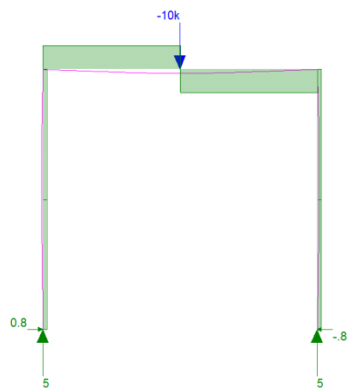
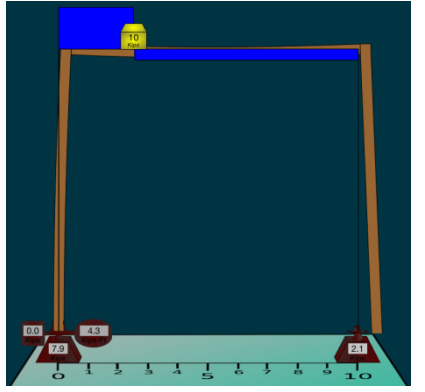
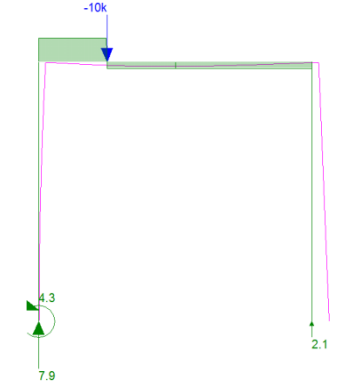
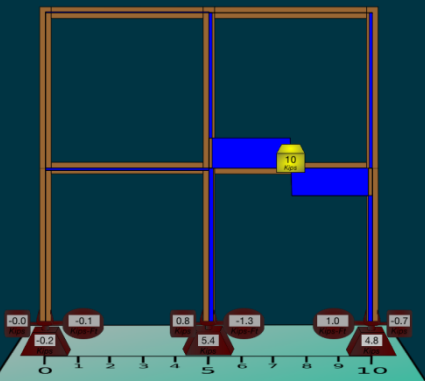
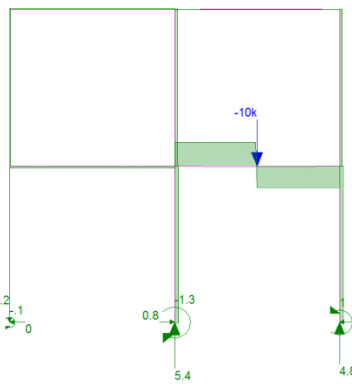
<i>iStructure</i> Shear Diagram	Point Load Center
 <p>The diagram shows a rectangular frame with a horizontal beam of length 10 and two vertical columns of height 5. A point load of 10 is applied at the center of the beam (x=5). The shear force is zero on the columns and constant at -0.8 on the beam. The supports are pinned, with reaction values of 0.8 at the base of each column.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The shear force is constant at 0.8 on the left column and -0.8 on the right column. The beam shear force is zero.</p>
Fully Pinned End Frame	
 <p>The diagram shows a rectangular frame with a horizontal beam of length 10 and two vertical columns of height 7.9. A point load of 10 is applied at the center of the beam (x=5). The left column is fixed, and the right column is a roller. Reaction values are 0.0 and 4.3 at the top of the left column, 7.9 at the base of the left column, and 2.1 at the base of the right column.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The shear force is constant at 4.3 on the left column and 2.1 on the right column. The beam shear force is zero.</p>
Fixed and Roller End Frame	
 <p>The diagram shows a two-bay frame with two vertical columns of height 5.4 and a horizontal beam of length 10. A point load of 10 is applied at the center of the beam (x=5). The left column is fixed, and the right column is a roller. Reaction values are -0.0, -0.1, 0.8, -1.3, 1.0, and -0.7 at the top of the columns, and -0.2, 5.4, and 4.8 at the bases.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The shear force is constant at 0.8 on the left column and -0.7 on the right column. The beam shear force is zero.</p>
Fully Fixed End Frame	

Table 23: Frame Shear Diagram with SAP2000 Results for Lateral Loads

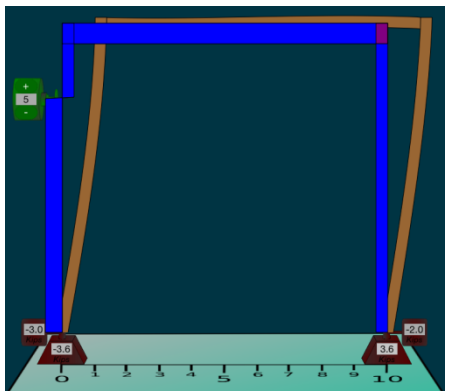
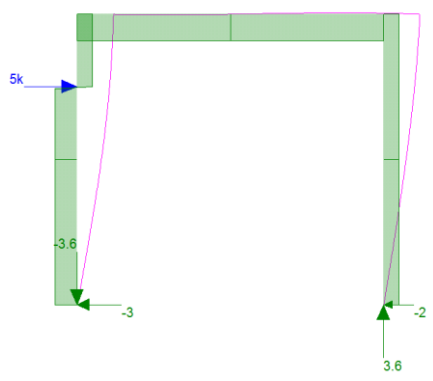
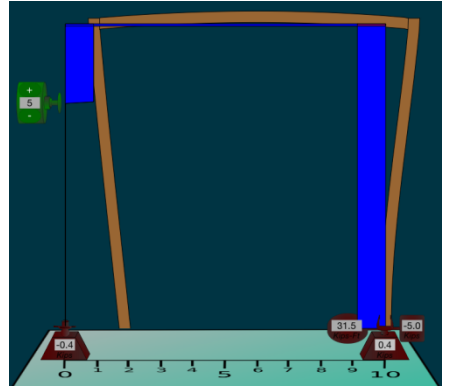
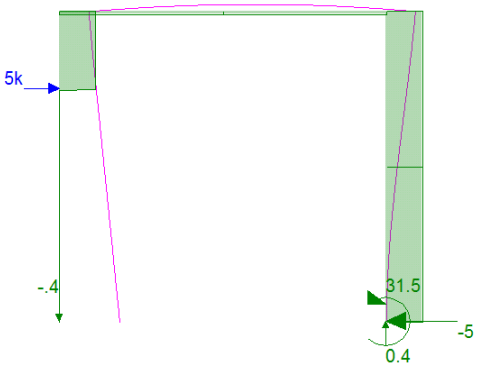
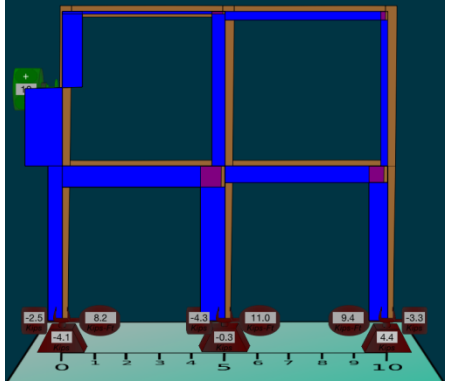
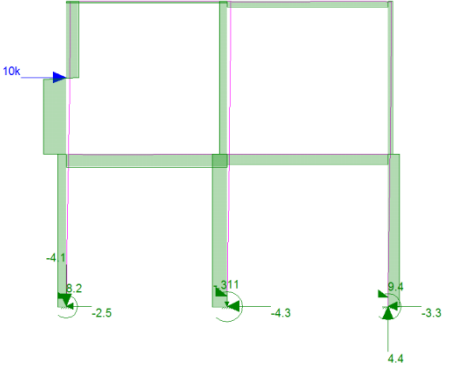
<i>iStructure</i> Shear Diagram	Lateral Load
 <p>The diagram shows a frame with two vertical columns and a horizontal beam. The left column is pinned at the base, and the right column is also pinned. Shear values are indicated: +5 on the left column, -3.0 at the top-left corner, -3.6 at the base of the left column, +2.0 at the top-right corner, and +3.6 at the base of the right column. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows a frame with a 5k lateral load applied to the left column. Shear values are indicated: -3.6 at the base of the left column, -3 at the top-left corner, -2 at the top-right corner, and +3.6 at the base of the right column.</p>
Fully Pinned End Frame	
 <p>The diagram shows a frame with a pinned support on the left and a fixed support on the right. Shear values are indicated: +5 on the left column, -0.4 at the top-left corner, -31.5 at the base of the left column, +0.4 at the top-right corner, and -5.0 at the base of the right column. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows a frame with a 5k lateral load applied to the left column. Shear values are indicated: -4 at the base of the left column, 31.5 at the base of the right column, and -5 at the top-right corner. A rotation of 0.4 is shown at the base of the right column.</p>
Pinned and Fixed End Frame	
 <p>The diagram shows a frame with fixed supports at both ends. Shear values are indicated: -2.5 at the top-left corner, -4.1 at the base of the left column, +8.2 at the top-right corner, -4.3 at the base of the middle column, +11.0 at the top-left corner of the right column, -9.4 at the top-right corner, +4.4 at the base of the right column, and -3.3 at the top-right corner. The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows a frame with a 10k lateral load applied to the left column. Shear values are indicated: -4.1 at the base of the left column, +8.2 at the top-right corner, -3.11 at the base of the middle column, -4.3 at the top-right corner, +9.4 at the top-right corner, and -3.3 at the base of the right column. A rotation of 4.4 is shown at the base of the right column.</p>
Fully Fixed End Frame	

Table 24: Frame Moment Diagram with SAP2000 Results for Center Point Loads

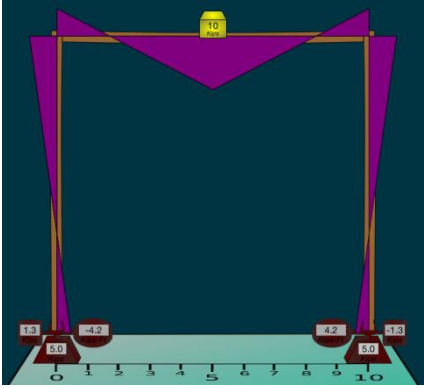
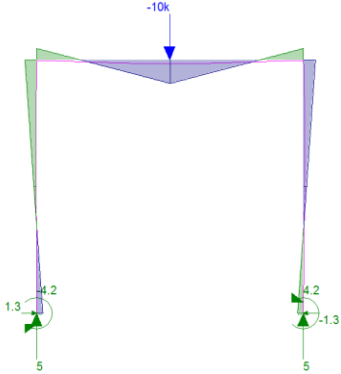
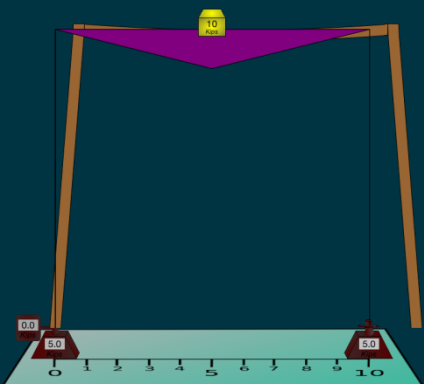
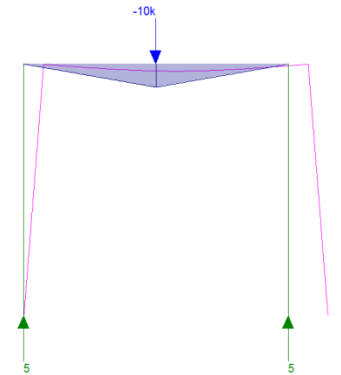
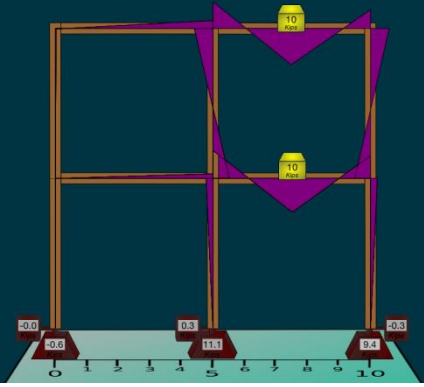
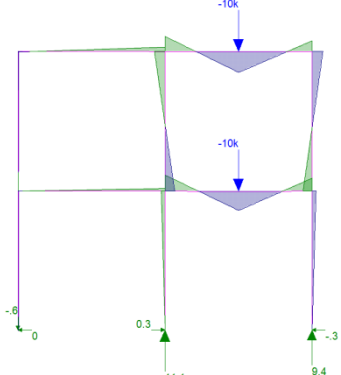
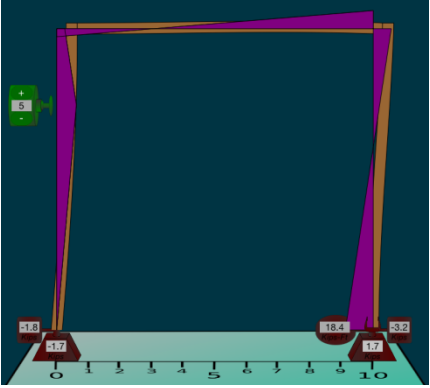
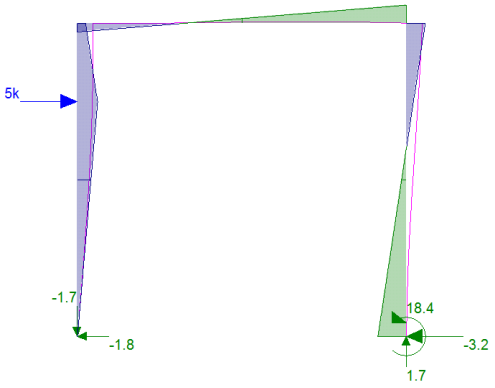
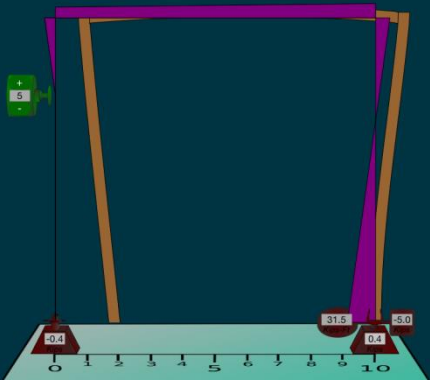
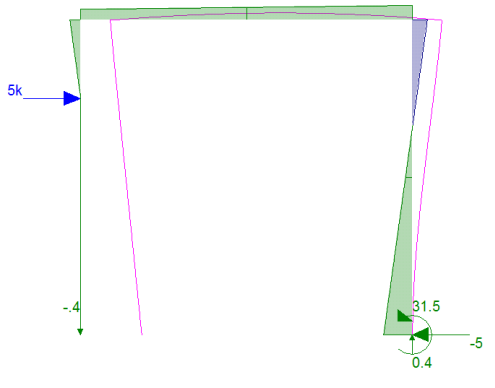
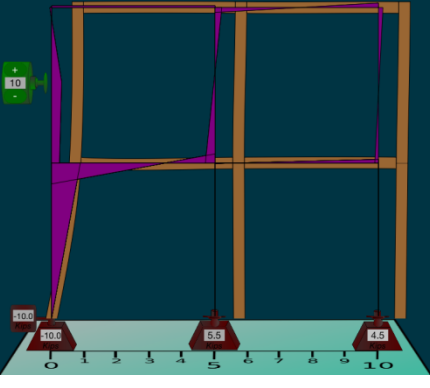
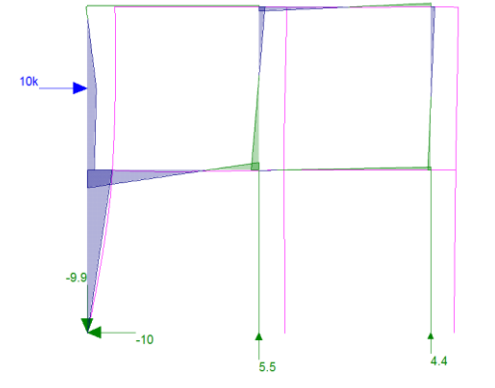
<i>iStructure</i> Moment Diagram	Point Load Center
 <p>The diagram shows a rectangular frame with a horizontal beam of length 10 units and two vertical columns of height 5 units. A point load of 10 k is applied at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is 10 k. At the base of each column, the moments are 4.2 k (top) and 1.3 k (bottom). The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is -10k. At the base of each column, the moments are 4.2 k (top) and -1.3 k (bottom). The horizontal axis is labeled from 0 to 10.</p>
Fully Fixed End Frame	
 <p>The diagram shows a rectangular frame with a horizontal beam of length 10 units and two vertical columns of height 5 units. A point load of 10 k is applied at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is 10 k. At the base of each column, the moments are 0.0 k (top) and 5.0 k (bottom). The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is -10k. At the base of each column, the moments are 0.0 k (top) and 5.0 k (bottom). The horizontal axis is labeled from 0 to 10.</p>
Pinned and Roller End Frame	
 <p>The diagram shows a rectangular frame with a horizontal beam of length 10 units and two vertical columns of height 5 units. A point load of 10 k is applied at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is 10 k. At the base of each column, the moments are 0.0 k (top) and 9.4 k (bottom). The horizontal axis is labeled from 0 to 10.</p>	 <p>The diagram shows the same frame with a point load of -10k at the center of the beam. The moment diagram is shaded purple. At the top of the beam, the moment is -10k. At the base of each column, the moments are 0.3 k (top) and -9.4 k (bottom). The horizontal axis is labeled from 0 to 10.</p>
Fully Pinned End Frame	

Table 25: Frame Moment Diagram with SAP2000 Results for Lateral Loads

<i>iStructure</i> Moment Diagram	Lateral Load
 <p>The diagram shows a frame with a horizontal span of 10 units. A lateral load of 5k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is -1.8. At the bottom-right corner, the moment is 18.4. At the top-left corner, the moment is 17.7. At the top-right corner, the moment is 3.2. A scale bar at the bottom indicates a maximum moment of 5.</p>	 <p>The diagram shows a frame with a horizontal span of 10 units. A lateral load of 5k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is -1.7. At the bottom-right corner, the moment is 18.4. At the top-left corner, the moment is -1.8. At the top-right corner, the moment is 1.7. A scale bar at the bottom indicates a maximum moment of 5.</p>
Pinned and Fixed End Frame	
 <p>The diagram shows a frame with a horizontal span of 10 units. A lateral load of 5k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is 0.4. At the bottom-right corner, the moment is 0.4. At the top-left corner, the moment is 31.5. At the top-right corner, the moment is 5.0. A scale bar at the bottom indicates a maximum moment of 5.</p>	 <p>The diagram shows a frame with a horizontal span of 10 units. A lateral load of 5k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is -4. At the bottom-right corner, the moment is 31.5. At the top-left corner, the moment is 0.4. At the top-right corner, the moment is 5.0. A scale bar at the bottom indicates a maximum moment of 5.</p>
Roller and Fixed End Frame	
 <p>The diagram shows a frame with a horizontal span of 10 units and a vertical height of 10 units. A lateral load of 10k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is -10.0. At the bottom-right corner, the moment is 4.5. At the top-left corner, the moment is 10.0. At the top-right corner, the moment is 5.5. A scale bar at the bottom indicates a maximum moment of 10.</p>	 <p>The diagram shows a frame with a horizontal span of 10 units and a vertical height of 10 units. A lateral load of 10k is applied at the top-left corner. The moment diagram is color-coded: purple for positive moments and green for negative moments. At the bottom-left corner, the moment is -9.9. At the bottom-right corner, the moment is 4.4. At the top-left corner, the moment is -10. At the top-right corner, the moment is 5.5. A scale bar at the bottom indicates a maximum moment of 10.</p>
Pinned and Roller End Frame	

7.3.3 Stress and Strain Diagram

Section 5.2.4 describes the direct correlation between the strain and bending moment. Therefore, having confidence in the bending moment process provides further confidence in the stress and strain diagram. The stress and strain validations are shown in Table 26 and Table 27. As you can see in Table 26, the stress and strain matches the behavior of the moment diagram.

Table 26: Frame Stress and Strain Validation with Moment Diagram

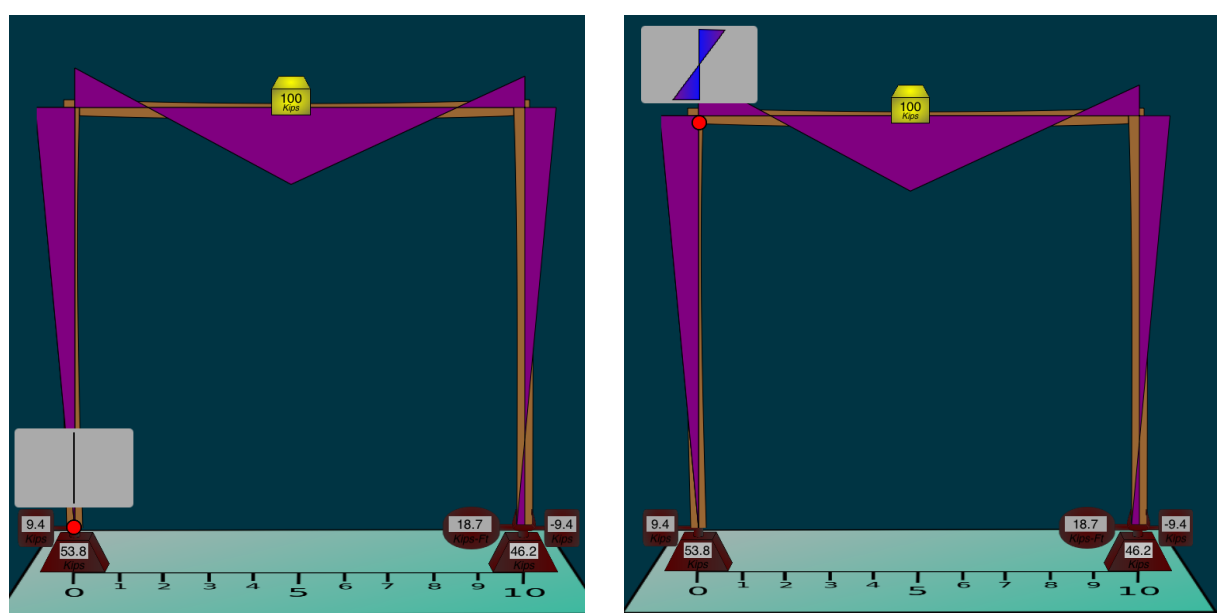
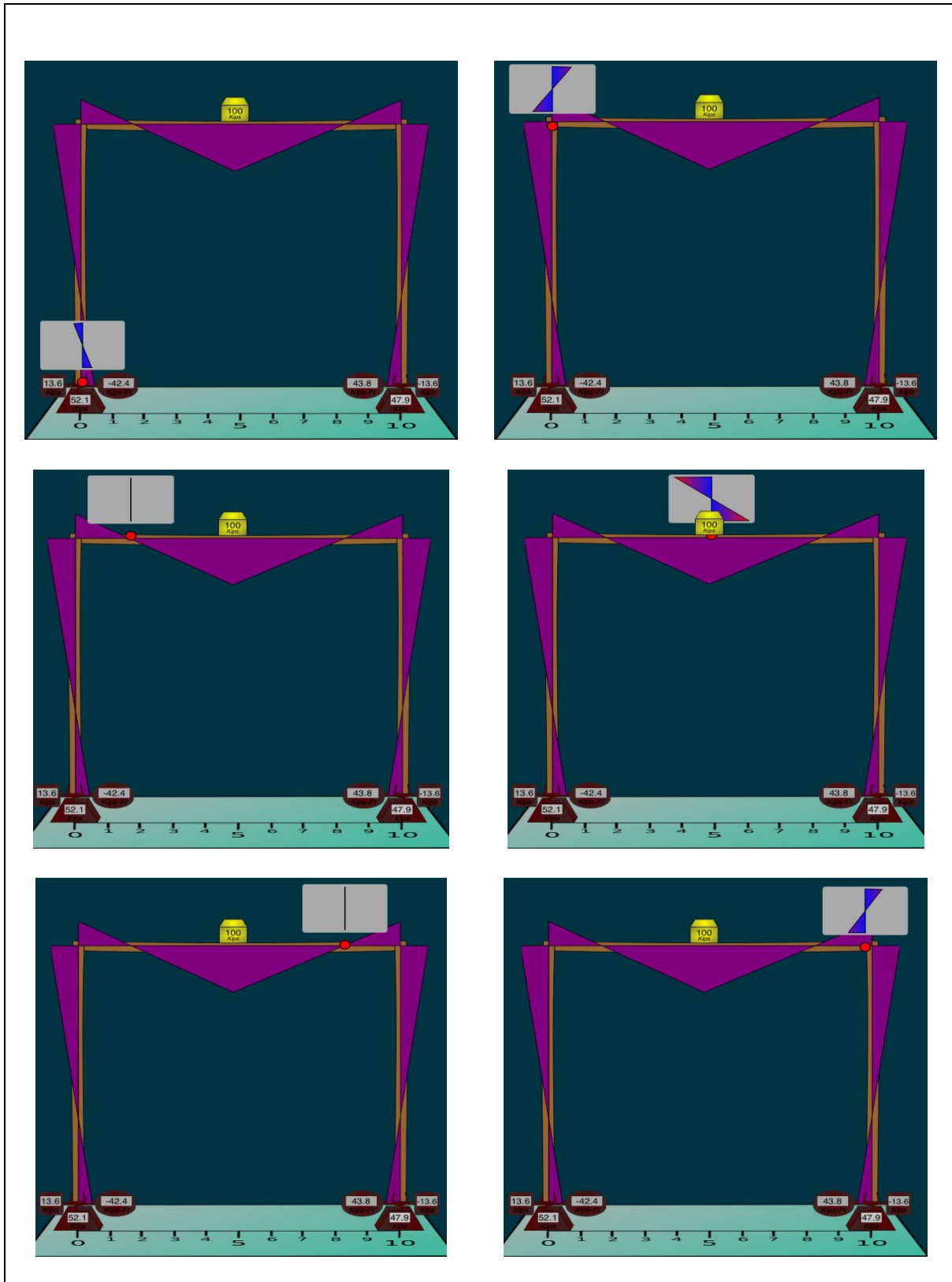


Table 27: Frame Stress and Strain Validations



Chapter 8: Summary and Conclusions

8.0 Summary

iStructure is an iPad application that provides users a visual representation of a structure's behavior based on Matrix Structural Analysis. The uniqueness of *iStructure* is its user-interactive features and structural animations, where structural configurations and loading conditions can be easily created and analyzed instantly, in an animated structure. The application consists of three modules: Beam, Truss, and Frame Module. The structural characteristics, such as structural deformation, internal force diagrams, stress and strain diagrams, and influence line are illustrated and simulated within each module using Objective-C programming.

The Frame and Truss Modules employ Matrix Structural Analysis methods for determining the conditions of the structure. The modules incorporate a modified and efficient matrix structural analysis procedure through the use of pointer addresses, which avoid developing multiple large matrices for structural analysis by using the DOF indices to develop the global partition matrix directly. This procedure provides faster computation and less memory demands, while making instant structural analysis of relatively complicated structural configurations possible.

The Beam Module utilizes a predefined beam formulas method for determining the conditions of the structure. This method significantly reduces the structural analysis computation demand which is necessary for the development of the Influence Line, which requires multiple structural analyses iteration.

8.1 Conclusions

The Matrix Structural Analysis method for this application provides an effective routine for analyzing and animating structural behavior continuously without strenuous demand on the iPad device. This method implements a pointer address technique in place of the traditional local member matrix and global matrix operations. The pointer address technique determines the desired unconstrained DOF and develops the partitioned global matrix directly, as opposed to developing multiple matrices and performing multiple matrix operations. This Matrix Structural Analysis method is used to efficiently analyze determinate and indeterminate structures and to develop the structural deformation.

The internal force diagram function for the Truss and Frame Module is achieved by developing the local force matrix using the structural deformation and structural stiffness relationship. The stress and strain diagram function for this module is achieved by developing the total strain using the structural deformation and internal force matrixes. The internal force diagram and stress and strain diagram can be successfully applied in the iPad application.

The beam deflection and internal force diagram is calculated by using predetermined beam formulas. This technique was adopted, in lieu of the Matrix Structural Analysis method, to further reduce the computational demand and to be more efficiently computational for the iPad application. The stress and strain diagram is developed by using the internal force diagram that is developed from the predetermined beam formulas. For continuous beam conditions, the Three Moment method can be effectively employed in the structural beam analysis function. Due to the Beam Module's low computation demand, multiple beam structure analyses iterations can be performed in order to develop an influence line diagram without the user experiencing computational lag. This Influence Line function can provide reaction influence line for any supports, and can determine the influence line for shear forces and bending moments for any point along the beam structure. The efficiency of the Beam Module computation can allow for immediate user interaction and animation that other structural analysis software may not be able to provide nor applicable for iPad applications.

8.2 Recommendations

Advances and additions to the *iStructure* application may consist of influence line functions for the Truss and Frame Module, which may need to be developed as a one-time computed feature rather than a continuous and animated feature. This is due to the computational demand of the matrix structural analysis and the intricate unit load path of the influence line for complicated structures, such as multi-story and bay frames. The application could be enhanced by including a function to allow the user to input material properties and return a numerical solution rather than conceptual solutions. Since the application employs object oriented programming methods, the structure members and nodes are created as custom objects and can retain unique properties. This allows alternation of individual member properties and nodes with easy data management, such as storing individual structural member properties.

References

- American Institute of Steel Construction. (2011). *AISC 14th Ed.*
- Apple Inc. (2010). *Object-Oriented Programming with Objective-C*. Retrieved 10 8, 2014, from Mac Developer Library:
https://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/OOP_ObjC/Articles/ooObjectModel.html#//apple_ref/doc/uid/TP40005149-CH5-SW4
- Apple Inc. (2014). *iOS Dev Center*. Retrieved 2014, from Apple Developer:
<https://developer.apple.com/devcenter/ios/index.action>
- Cochrane, T. (2010, July). Exploring mobile learning success factors. *ALT, Research in Learning Technology*, 133-148.
- Computers and Structures, Inc. (2014). SAP2000.
- Davalos, J. F., & Moran, C. J. (1996, Spring). Laboratory Manual for Structural Analysis I. Morgantown, West Virginia, United States of America.
- Ebner, M., & Holzinger, A. (2007). Successful implementation of user-centered game based learning in higher education: An example from civil engineering. *Computers & Education* 49, 873-890.
- Ebner, M., & Walder, U. (2008). E-Education in Civil Engineering - A Promise for the Future. *6th AECEF Symposium on Civil Engineering Education* (pp. 16-26). Changing Europe: VGTU Press.
- Gere, J. M., & Timoshenko, S. P. (1997). *Mechanics of Materials*. PWS Publishing Company.
- Herrington, A., & Herrington, J. (2007). Authentic mobile learning in higher education. *AARE Annual Conference*. Fremantle.
- Kassimali, A. (2012). *Matrix Analysis of Structures*. Stamford: Cengage Learning.
- RISA Technologies, LLC. (2014). RISA-2D.

Appendix A: Xcode Object Classes

iStructure is developed using Xcode, an integrated development environment (IDE) Apple software. The primary language used to code *iStructure* is Objective-C, which is subpart of the C-family code. Various techniques are used to produce an interactive and representative environment, such as the matrix structural analysis methods which are used to solve indeterminate structures and the loop cycles which simulate the environment's gravity and object interaction within the *iStructure* Application.

Common Xcode Object Class

C-Style Array/Matrix	Used for creating instant variable arrays and matrices without the need to retain memory. (Instant Variable)
Float	Used for holding a finite numerical value. (Instant Variable)
NSMutableArray	NSObject Class that defines the properties and methods for a mutable array object. Is able to be rearranged and add/remove object after initialization (mutable). Used for retaining and sending objects.
NSObject	Root Object Class for most object classes defined in the Objective-C language.
Objective-C Class File	File in which object classes and methods are written in.
Objective-C Category File	File used to hold additional class methods and functions. Variable instances cannot be declared within the file.
UIBezierPath	An object that defines a Bezier path within the UIView Object. Can be used to draw and display elements on current UIView Object.
UIImageView	An object that defines an image's view. Allows an image to be altered through its shape and appearance.

UIView

An object that defines the properties and methods for a view object which can be added to the user-interface subview. Is used to manually draw and display images onto the user-interface within the UIView's frame.

UIViewController

An object that defines the properties and methods for a view controller object. UIViewController objects represent the modules for the application. Contains object instances, methods, and defines the user-interface subview.

Appendix B: iStructure Object Classes

Module Objects

Module	<i>UIViewController</i>	Contains and initiates object instances. Defines the custom methods for structural analysis and user-interaction for each structure module, respectively.
StructuralView	<i>UIView Class</i>	Contains the structure's properties and defines methods for drawing deformed and non-deformed structures.
StrainStressView	<i>UIView</i>	Contains the strain's properties and defines methods for drawing the stress and strain profile.
Load Objects	<i>Custom Class</i>	Contains the properties for an applied load, including: location, magnitude, images, and etc.
Support Objects	<i>Custom Class</i>	Contains the properties for a support element, including: location, support reactions, constraint, images, and etc.
AppliedLoadMatrix	<i>NSMutableArray</i>	Contains and retains the load objects. Used for managing and sending the load objects.
SupportMatrix	<i>NSMutableArray</i>	Contains and retains the support objects. Used for managing and sending support objects.

Matrix Structural Objects and Variables

Angle Array	<i>C-Style Array</i>	Contains the member angles.
Fixed End Force Array	<i>NSMutableArray</i>	Contains and retains the nodal equivalent forces. Used for sending values as a collective.
Global Forces	<i>NSMutableArray</i>	Contains and retains the nodal global forces. Used for managing and sending values as a

		collective.
Joint Deflection	<i>NSMutableArray</i>	Joint (nodal) deflection solution obtained from Matrix Structural Analysis function. Used for sending values as a collective.
JointMatrix	<i>C-Style Matrix</i>	Contains the joint's index matrix, constraint matrix, and joint deflection matrix.
Length Array	<i>C-Style Array</i>	Contains the member lengths.
Local Forces	<i>NSMutableArray</i>	Contains the nodal local forces. Used for managing and sending values as a collective.
Partitioned Global Stiffness Matrix	<i>C-Style Matrix</i>	Contains the partitioned global stiffness matrix. Used for determining the joint (nodal) deflection.
Partitioned Stiffness Pointer Array	<i>C-Style Array</i>	Contains the index positions of DOFs in a partitioned stiffness matrix. Used for directly developing the partitioned stiffness matrix.

StructureView Objects

MemberBound	<i>UIBezierPath</i>	Used for drawing a member's path on the parent UIView in the UIView's local coordinates.
MemberBounds	<i>NSMutableArray</i>	Contains and retains the drawn MemberBound objects. Used for managing and sending UIBezierPath objects as a collective.
StructureBound	<i>UIBezierPath</i>	Object Path for which the MemberBound paths are appended to. Used for drawing and displaying structure on the parent UIView in the UIView's local coordinates.
DiagramBound	<i>UIBezierPath</i>	Used for drawing and displaying the diagram on

the parent UIView in the UIView's local coordinates.

StrainStressView Objects

Beam Curvature	<i>Float</i>	Contains the beam curvature value. Used for developing the bending stain.
Axial Strain	<i>Float</i>	Contains a member's axial strain value. Used in developing the member's total strain.
StrainBound	<i>UIBezierPath</i>	Used for drawing and displaying the stress and strain profile path on the parent UIView in the UIView's local coordinates.

InfluenceLineView Objects

UnitLoad	<i>Custom Class</i>	Contains the properties for a unit load, including: location, magnitude, and etc. Excludes user visual and interaction properties.
InfluenceSupport	<i>Custom Class</i>	Contains the selected support object for the Reaction Influence Line function.
InfluencePoint	<i>CGPoint</i>	Contains a point coordinate along the beam structure, specified by the user for point of analysis.

Appendix C: iStructure Functions

Common Functions

<u>Function</u>	<u>Location</u>	<u>Description</u>
3-DOF Global Stiffness Function	<i>Category File</i>	Predefined stiffness matrix for a 3-DOF element in global coordinates. Requires i and j -index position and returns the corresponding matrix element value.
2-DOF Global Stiffness Function	<i>Category File</i>	Predefined stiffness matrix for a 2-DOF element in global coordinates. Requires i and j -index position and returns the corresponding matrix element value.
Beam Formulas	<i>Category File</i>	Predefined Euler-Bernoulli beam formulas for fixed, pinned, roller boundary conditions. Obtained from AISC manual. Used to develop the beam deflection, shear and moment diagrams.
Beam Formulas Shear Function	<i>Category File</i>	Beam formula for shear force for different beam conditions. Function is defined in terms of distance along the beam.
Beam Formulas Moment Function	<i>Category File</i>	Beam formula for moment force for different beam conditions. Function is defined in terms of distance along the beam.
Beam Formulas Reaction & Deflection	<i>Category File</i>	Beam formula for support reactions and beam deflection for different beam conditions. Function is defined in terms of distance along the beam for the beam deflection.
Draw Axial Diagram Member	<i>UIView</i>	Draws the axial diagram of the one member in the UIView Object. Requires diagram coordinate points stored in an NSMutableArray and

		member's global point location. Assigns drawing to a UIBezierPath objects.
Draw Member Function	<i>UIView File</i>	Draws one structural member in the UIView Object. Requires member's coordinate points stored in an NSMutableArray and member's global point location. Assigns drawing to a UIBezierPath objects.
Draw Moment Diagram Member	<i>UIView</i>	Draws the moment diagram of the one member in the UIView Object. Requires diagram coordinate points stored in an NSMutableArray and member's global point location. Assigns drawing to a UIBezierPath objects.
DrawRect	<i>UIView</i>	UIView object's default draw function. Can only be called indirectly by <i>setNeedsDisplay</i> function and only initiated at the end of runtime. Draws out the deformed and non-deformed structure image. Draws the axial, shear, bending moment diagrams.
Draw Shear Diagram Member	<i>UIView</i>	Draws the shear diagram of one member on the parent UIView window. Requires diagram coordinate points stored in an NSMutableArray and member's global point location. Assigns drawing to a UIBezierPath objects to be drawn.
Draw StrainStress Diagram	<i>UIView</i>	Draws the stress and strain diagram at the Point of Interest on the parent UIView window. Requires diagram coordinate points stored in an NSMutableArray and member's global point location. Assigns drawing to a UIBezierPath objects to be drawn.

Gauss Elimination	<i>Category File</i>	Used for solving the joint (nodal) deflections. Requires the stiffness matrix and equivalent nodal forces in one NSMutableArray.
Gravity Function	<i>UIViewController</i>	Moves falling objects a specified distance per cycle. Defines object-to-object interaction.
Main Run Loop Function	<i>UIViewController</i>	Initiates and cycles the modules functions per runtime.
Matrix Structural Analysis Function	<i>Category File</i>	Used for analyzing a structure for different structural configurations.
Three Moment Equation	<i>Category File</i>	Used for determining the reactions of the interior supports of a continuous beam.
Transformation Matrix Function	<i>Category File</i>	Used for transforming the local coordinate based elements to global coordinates.
Structural Analysis	<i>UIViewController</i>	Initiates the analysis of the structure and determines the structural configuration required for the Matrix Structural Analysis and Beam Formulas function.