東京大学大学院新領域創成科学研究科
社会文化環境学専攻

2020 年度
修　士　論　文

鋼製骨組にガラス板が拘束された構造体における
初期剛性を再現するモデル化

Modelling Method for Initial Stiffness of the Glass Panels
Stiffened by Steel Frame

2020 年 1 月 20 日提出
指導教員　佐藤　淳　准教授

大霜 潤也
Oshimo, Junya

目次

第1章 序論

1.1 研究の背景

ガラスは紀元前 4000 年ほど前に発見されてから現在に至るまでに様々な使われ方をしてきた。初期は装飾品や皿などのガラス器として使われていたが、吹きガラス技法により小さな板ガラスを作ることが可能になったことで、ガラスは建築の分野へも利用の幅を広げていくこととなった。その代表例として挙げられるのがステンドグラスである。ステンドグラスは 12 世紀から中世ヨーロッパで発展したゴシック建築と共に発展し、フランスのシャルトル大聖堂やサントシャペルのように後に世界遺産として登録される建築も数多く生まれた。

19 世紀に入るとガラス工業が発展し、鉄骨とガラスを用いたクリスタルパレス（水晶宮）に代表されるように、ガラスは窓ガラス、さらには望遠鏡や顕微鏡といった光学用のガラスとしても利用されるようになった。

現在、ガラスは建築での利用が大部分を占めるが、建築の分野以外でも車や生活用品、電気製品といった様々な分野で利用されており、ガラスの持つ透明性や強度といった特性が随所に活かされている。[1)]

さて、生産されるガラスの大部分は建築に利用されており、その主な利用先はカーテンウォールや開口部、外装材などである。これらの非構造部材としての利用においては、ガラスの透光性や透視性、反射性といった特性が発揮された建築が数多くみられる。一方で、ガラスの持つ剛性や強度を活かし、構造部材としてガラスが用いられている例は少ない。現在、構造部材として用いられている耐力壁は不透明な構造体であるが、透明なガラスが構造部材として利用可能になると、より透明感のある空間を創り出すことができ、建築表現の幅が大きく広がると考えられる。また、ガラスが構造部材として働くため、鉄筋などの他の構造部材の材料使用量が削減され、環境の面でも意義があると考えられる。

1.2 ガラスの特性

先に述べたようにガラスはその剛性や強度に優れた特性を有している。ガラスのヤング率は 71,000N/㎟であり、アルミニウムの 70,000N/㎟とほぼ同等の性能であるとともに、コンクリートの 20,500N/㎟の約 3 倍の性能を持つ。したがって、ガラスは構造部材として利用可能な性能を十分持っていると言える。[2),3)]ガラスと他の材料の性能を比較したものを表 1.1 にまとめる。

表 1.1 ガラスと様々な材料の性能比較(単位:N/㎟)

材質	ヤング係数	圧縮強度	引張強度	曲げ強度
鋼鉄(SS400)	205,000	400	400	400
アルミニウム合金	70,000	-	240	-
普通ガラス	71,000	900	56	60
強化ガラス	71,000	-	-	360
普通コンクリート	20,500	40	3	-

本研究では、鋼製骨組にガラス板が拘束された構造体（以後「ステンドグラス構造体」と呼ぶ。）を提案し、ガラスを構造部材として利用可能にすることを目指す。

1.3 ステンドグラス構造体概要

本研究で扱うステンドグラス構造体の概要について述べる。この構造体は小径の H 型鋼のフランジ溝にガラスをはめ込んだ構造体であり、骨組とガラスが直接接触することによる割れを防ぐために H 形鋼とガラスの間には錫が緩衝材として用いられている。力学的には H 形鋼とガラスが互いに座屈を補剛しあうことで剛性を発揮するものとなっている。

1.4 既往研究

佐藤らにより研究が行われてきたステンドグラス構造体は図 1.1 のような骨組の目地が通った模様にガラスを配置するよりも、図 1.2 のように骨組の目地が通らない模様（以後「卍型」と呼ぶ。）に配置した方が、ガラスが鋼製骨組の面外方向への座屈を拘束し、最大耐力が大きくなることが分かっている。また、図 1.3 のようにガラスと鋼製骨組の間に錫を緩衝材として挟むことで、ガラスが骨組と直接接触することによる割れを防ぐことができ、ガラスがその剛性を発揮できることが分かっている。したがって、図 1.2 に示す卍型にガラスを配置し、ガラスの外周に錫を緩衝材として用いた試験体に関しての研究が主として行われてきた。

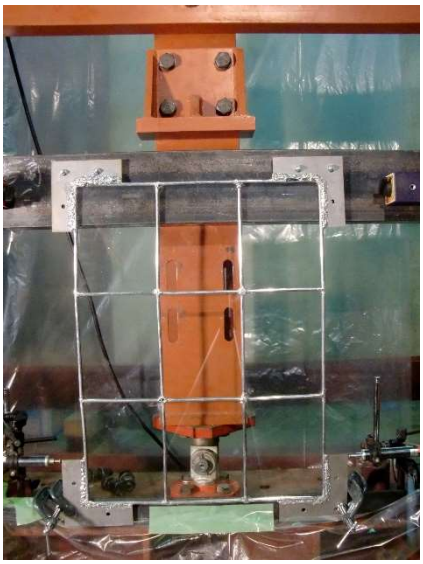


図 1.1　3×3 の試験体

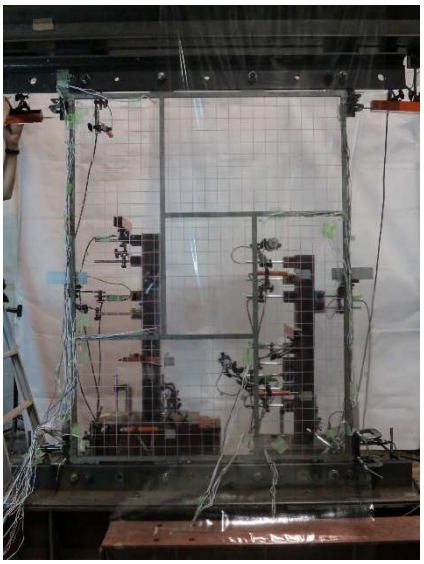


図 1.2　卍型の試験体

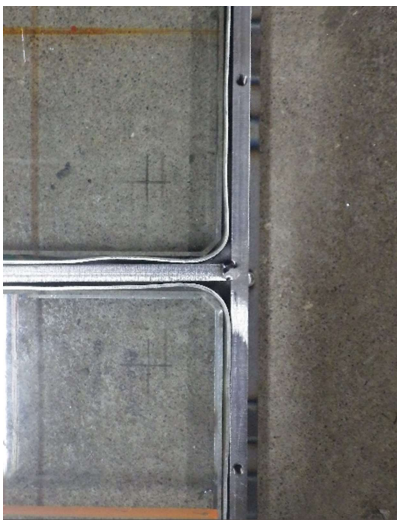


図 1.3　骨組とガラスの間に錫が挟まれている様子

卍型試験体の形状は図 1.4.に示す形状であり、過去に行われた試験体の種別を表 1.2 に示す。骨組の工法において、角鋼から削り出すことで H 形鋼を生成する手法のことを「削り出し」といい、フランジとウェブを別で作成して溶接とビス留めにより H 型鋼状に組み立てる手法を「組み立て」という。また、H 形鋼同士の交差部を溶接することで卍型の模様に組み上げる作成方法を「溶接」といい、1 枚の鋼板から卍型の模様にフランジを作成する方法を「一体成形」という。

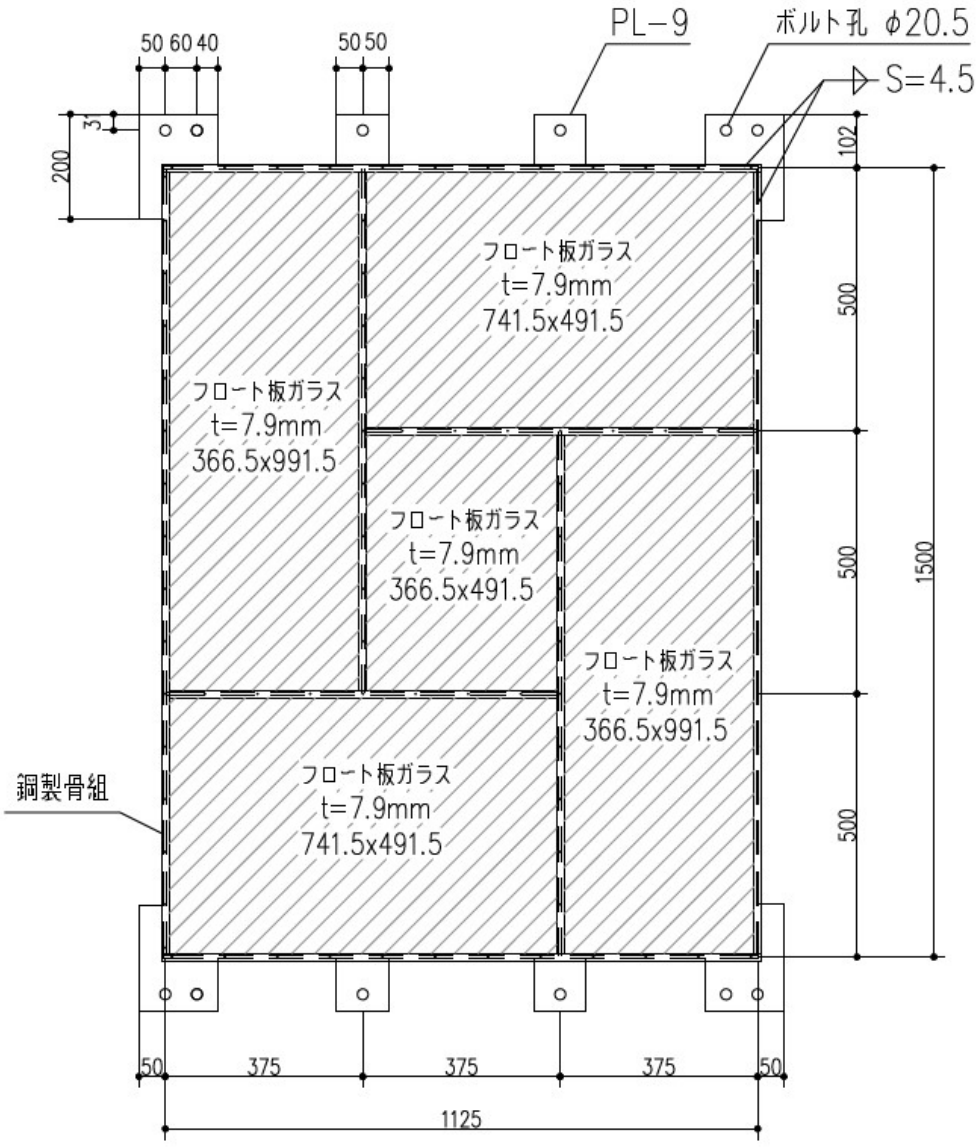


図 1.4　卍型試験体概形

表 1.2　既往卍型試験体形状まとめ

試験体名	フランジ幅 B(mm) H-15xBx4.5x3	骨組の工法	錫
卍型試験体 1	16x16x3x3	削り出し＋溶接	なし
卍型試験体 2	14	組み立て＋溶接	一部
卍型試験体 3	20	組み立て＋一体成形	あり 1 mm
卍型試験体 4	12		
卍型試験体 5	16		
卍型試験体 6	12		
卍型試験体 7	12		
卍型試験体 8	24		
卍型試験体 9	12		
卍型試験体 10	12		
卍型試験体 11	20		
卍型試験体 12	12		

既往研究において卍型試験体の解析手法は 2 通り試されている。1 つはガラスをブレース材として置換した簡易解析モデルである。この解析モデルでは、卍型試験体の上部に水平力が加わった際に、図 1.5 に示すように角の 2 等分線で囲まれた面積のガラスが剛性を発揮し、ブレースとして応力を負担するとしたものである。また、図 1.6 に示す手順でガラスをブレース材としてモデル化する。

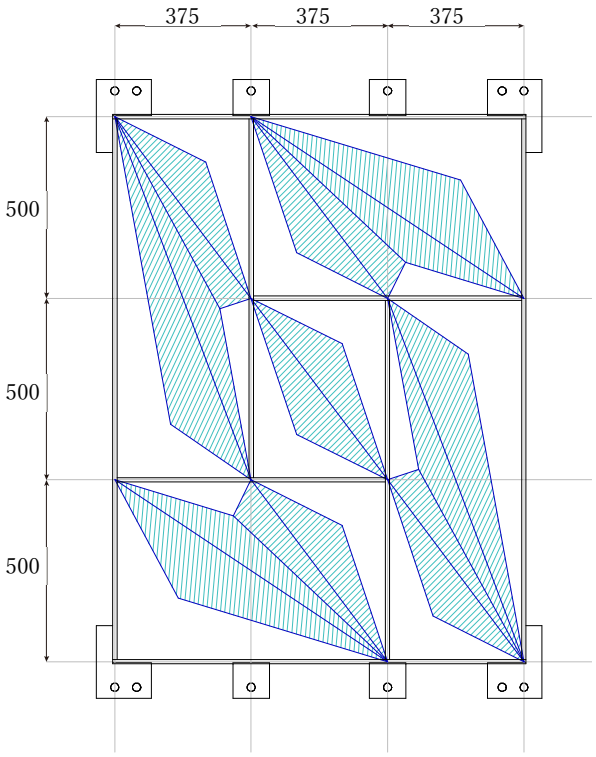


図 1.5　ガラスの応力負担範囲

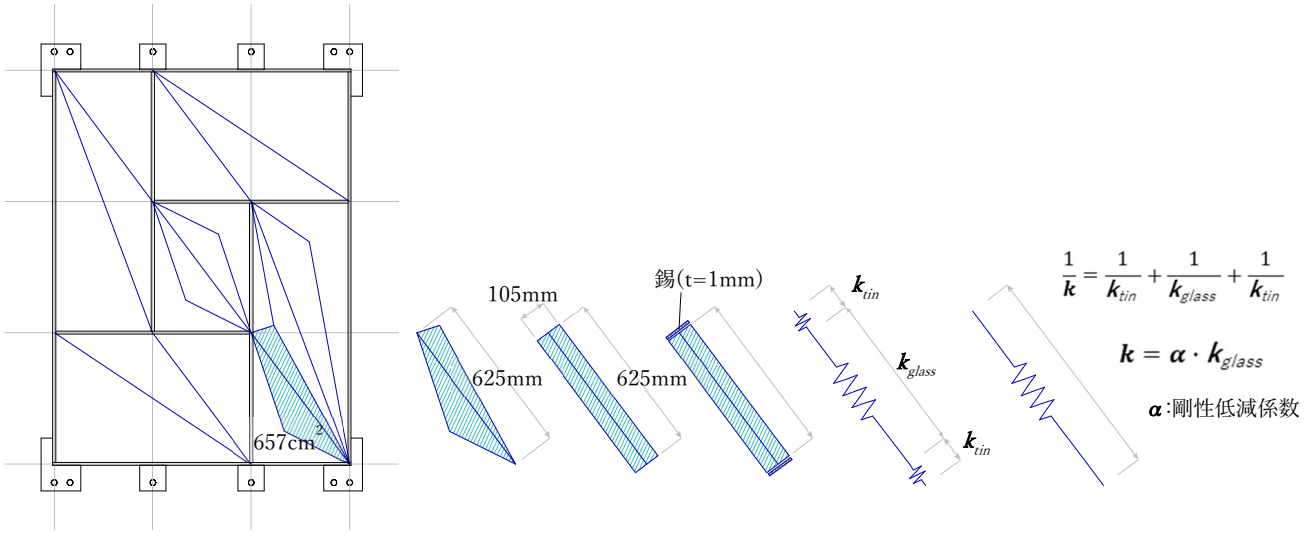


図 1.6　ガラスのブレース材置換手順

以下の図 1.7 に示す簡易解析モデルを用いて解析が行われ、ガラスの剛性の 1/20 倍の値をブレース材の剛性として入力すると、卍型試験体の載荷実験により得られる荷重変形曲線の初期剛性が再現されることが分かっている（図 1.8)。

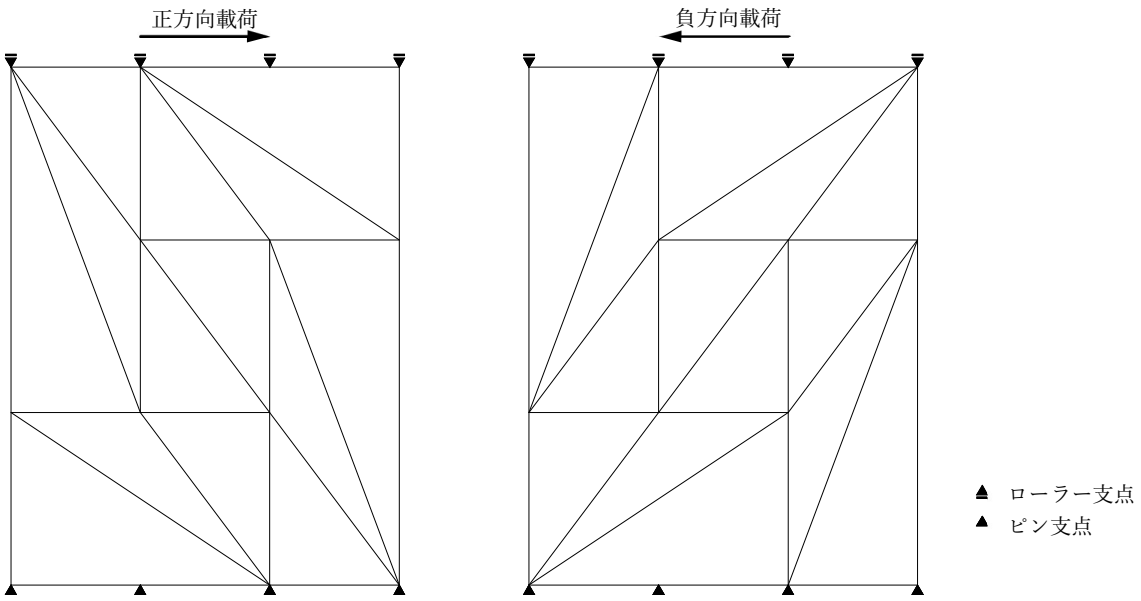


図 1.7　簡易解析モデル

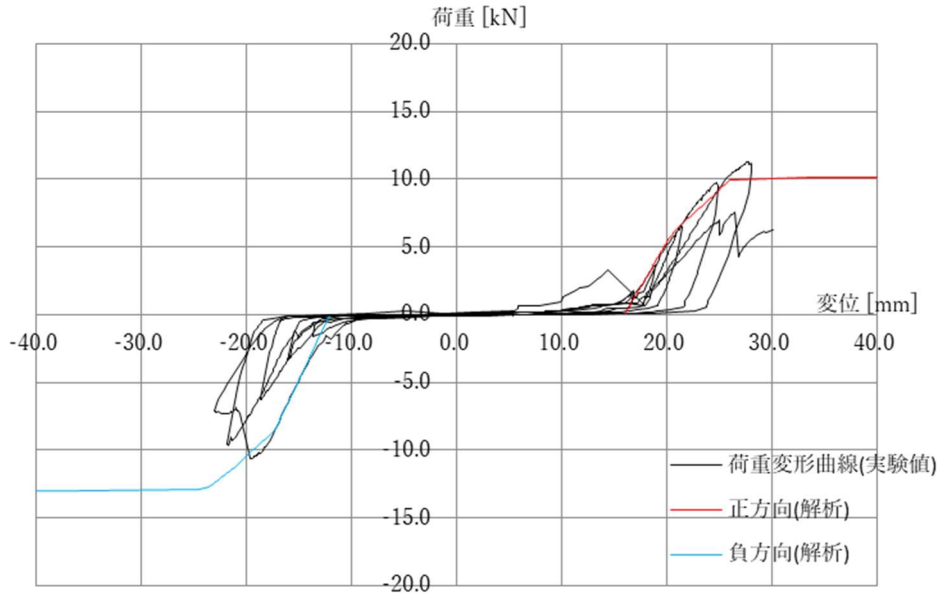


図 1.8　荷重変形曲線(実験値)と簡易解析モデルによる解析結果

もう 1 つはガラス、鋼製骨組、錫をメッシュ状に細分化した詳細解析モデル（図 1.9）である。この解析モデルは錫が錫ばねとしてモデル化されており、錫ばね部材がガラス部材と鉄骨部材のあいだ全周に入れられている（図 1.10）。

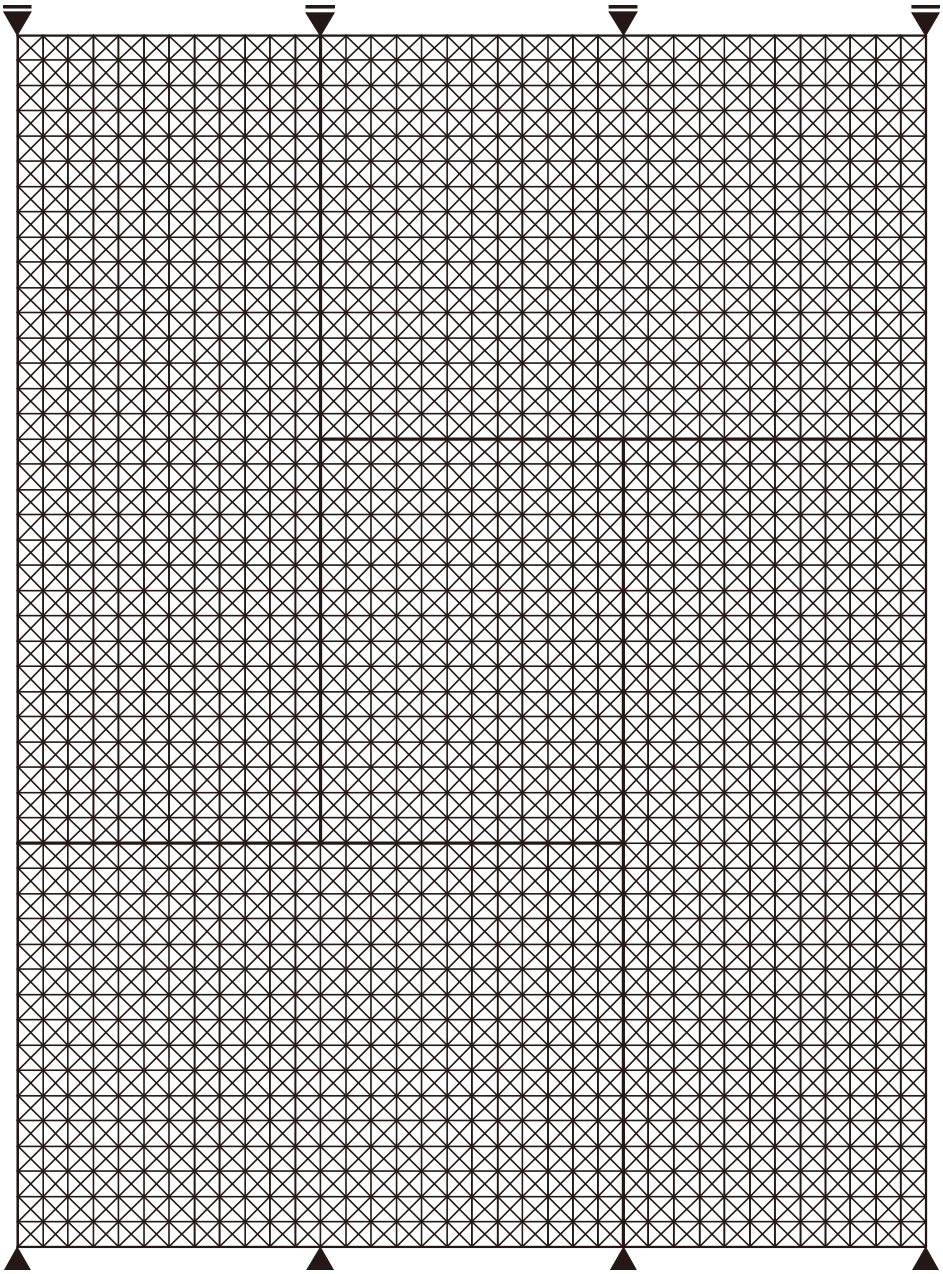


図 1.9　詳細解析モデル(全体図)

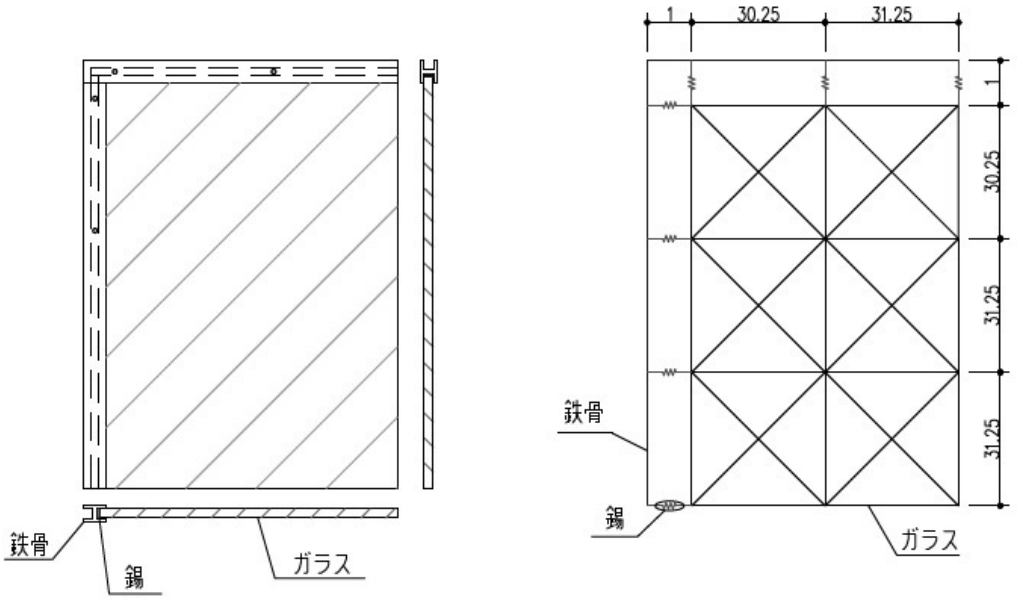


図 1.10　詳細解析モデル（部分拡大図）

また、錫ばねは引張を負担せず、圧縮のみを負担するものとし、解析は、線形解析を行ったのち引張の入った錫ばねを取り除き、再度線形解析を行う、という操作を引張の入った錫ばねがなくなるまで繰り返すという手法で行った（図 1.11)。この詳細解析モデルでは、錫ばねに働く応力がガラスの角の錫ばねほど大きくなり（図 1.12)、錫がガラスの角で押し潰される現象を解析に反映することができることが分かっているが、初期剛性は実験値より４倍大きい値となってしまう（図 1.13)。

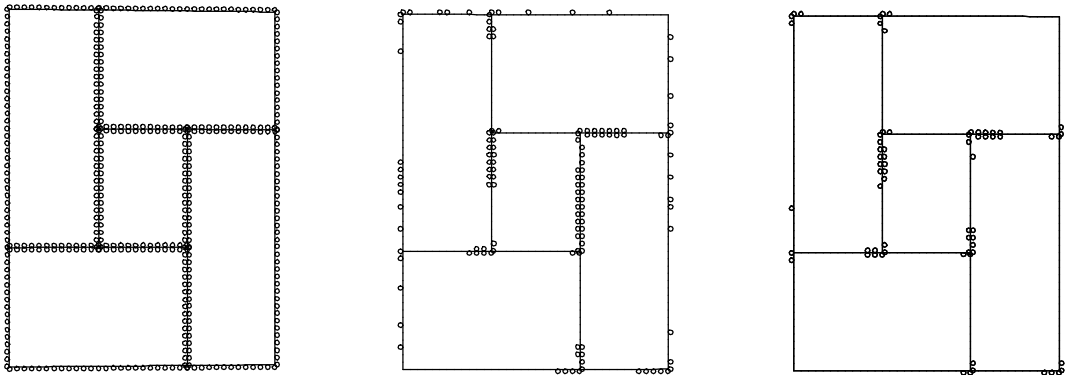


図 1.11　錫ばねが除去されていく様子

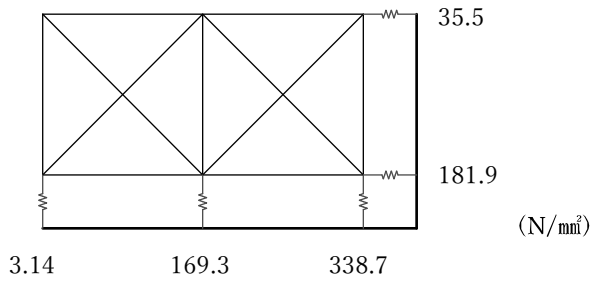


図 1.12　右下角部分の錫ばねの応力分布

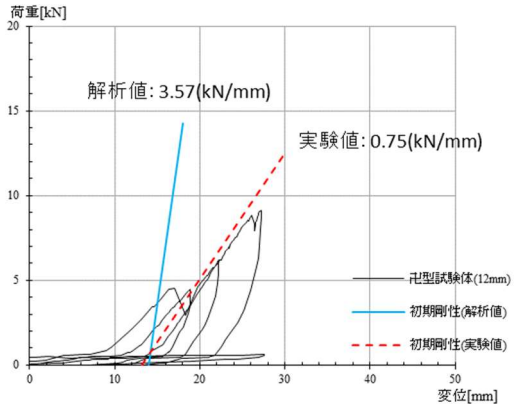


図 1.13　荷重変形曲線と詳細解析モデルによる初期剛性の関係

また、卍型試験体において初期剛性を再現するために、錫圧縮試験や、1 枚のガラス板を鋼製骨組で拘束した小型試験体の載荷試験が行われている。

錫圧縮試験（図 1.14）ではガラスの角で錫が圧縮される状態を再現した実験を行うことで、錫がガラスの角で押し潰される際の錫の剛性が得られる。小型試験体載荷試験（図 1.15）においては、ガラス材の端点に複数の錫ばねを設置し、長さに応じた剛性を錫ばねに入力する方法で解析を行うことで、小型試験体載荷試験により得られる荷重変形曲線のスリップ領域や初期剛性を再現できる可能性があることが分かっている。

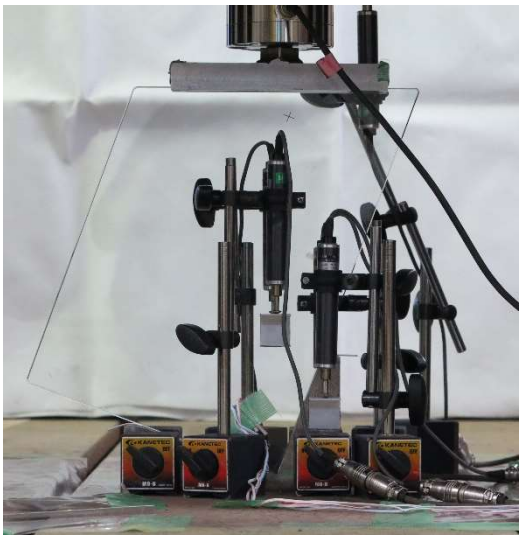


図 1.14　錫圧縮試験

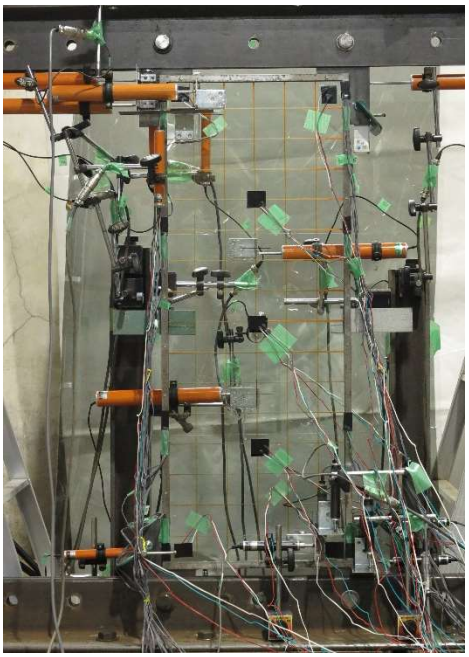


図 1.15　小型試験体載荷試験

1.5 本研究の目的及び本論分の構成

本研究では図 1.2 に示した卍型試験体の荷重変形曲線の初期剛性の大きさに影響を与えると予測される要素を解析モデルに反映させることで、初期剛性を再現することを目的としており、また、卍型以外の模様にも応用可能なモデル化の提案を行う。

本論文の構成を以下に述べる。

1 章　ガラスの剛性や強度に優れるという性質から、ガラスの構造部材として使用可能性について言及した。また、既往研究で得られた知見について述べ、本論文での研究目標について述べた。

2 章　ステンドグラス構造体の載荷試験について述べる。

3 章　モデル化で再現を試みる現象とそのモデル化の手法について述べる。

4 章　錫ばね除去法による解析を行った解析モデルとその結果について述べる

5 章　錫ばね剛性判定法による解析を行った解析モデルとその結果について述べる。

6 章　本論文での成果と今後の課題について述べる。

第2章 試験体載荷試験

2.1 概説

本研究では図 2.1 に示す卍型試験体の載荷試験を 2 体（卍型試験体 13、卍型試験体 14)、図 2.2 に示す小型試験体の載荷試験を 3 体(小型試験体 4、小型試験体 5、小型試験体 6)行った。

それぞれの試験体の形状や載荷方法、計測項目について次節以降にまとめる。また損傷状況や荷重変形曲線についても示す。

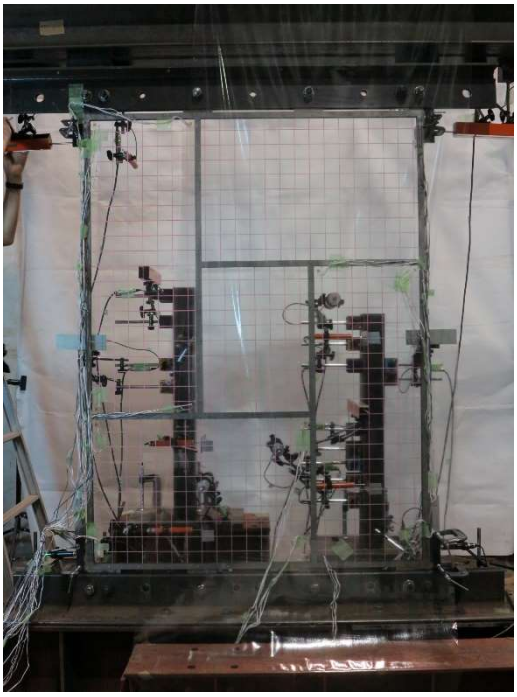


図 2.1 卍型試験体

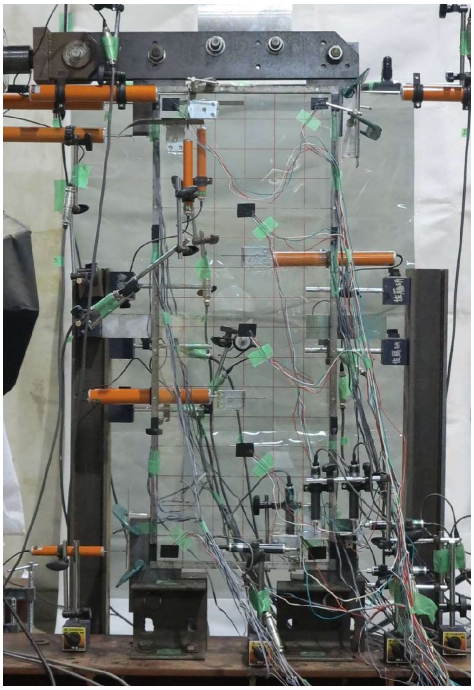


図 2.2 小型試験体

2.2 試験体作成

試験体の組み立て方法について述べる。

本研究で用いた試験体の鋼製骨組は全てフランジとウェブを別で作成し、溶接とビス留めにより H 形鋼状に組み上げられており、フランジは 1 枚の鋼板から各試験体の模様状に作成されている。したがって、まず初めに図 2.3、図 2.4 のようにフランジとウェブが溶接された T 字型の骨組材にガラスをはめ込む。次に図 2.5 のようにガラスと骨組材の間に錫を挿入する。そして最後に T 字型の骨組材と対になるフランジ材を重ね、ビスで留めることで試験体が組み立てられる（図 2.6、図 2.7）。

図 2.3　ガラスをはめ込んでいく様子

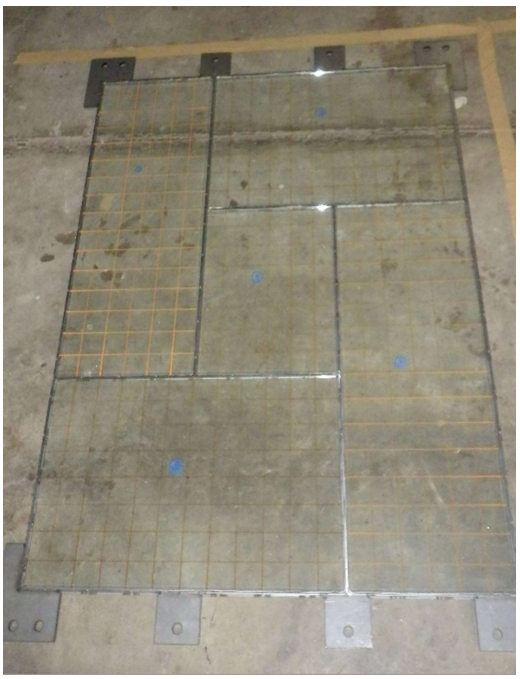


図 2.4　ガラスが全てはめ込まれた様子

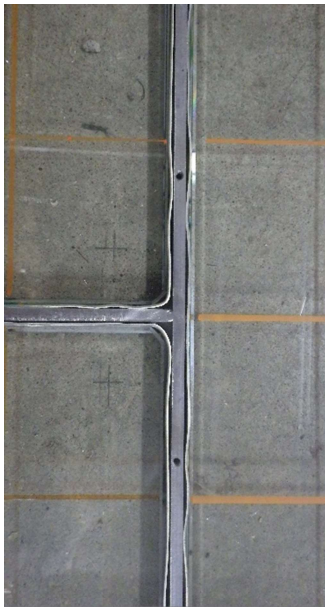
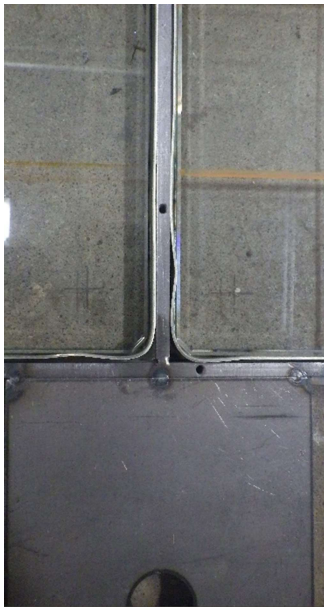
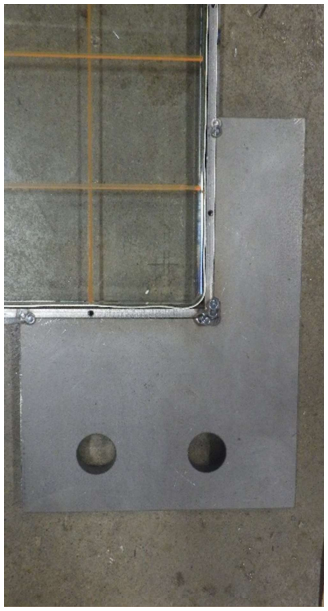




図 2.5　骨組とガラスの間に錫が挿入されている様子

図 2.6　フランジをビスで留める様子

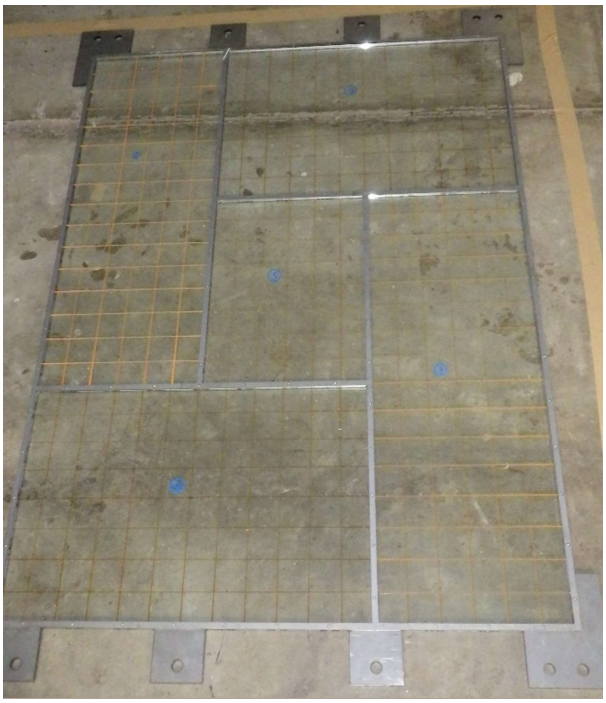


図 2.7　試験体組み立て完了

2.3 卍型試験体載荷試験

2.3.1 載荷試験概要

卍型試験体の載荷試験は H 型鋼製骨組のフランジ幅が 12mm（卍型試験体 13）と 16mm（卍型試験体 14）の 2 体行った。載荷方法は 2 体とも漸増とし、ガラスが破損し耐力がなくなったとみなした時点で載荷を終了した。

卍型試験体の寸法は鋼製骨組の芯寸法で[幅×高さ]が[1125mm×1500mm]であり、ガラスは厚さが t=7.9mm で 4 隅の角が 5mm 面取りされたフロート板ガラスを用いており、ガラスと鋼製骨組の間には厚さ t=1mm の錫を挿入している。

卍型試験体の形状は図 2.1、図 2.8 に示す形状であり、セットアップ図を図 2.9 に、計測器の設置位置および歪ゲージの貼り付け位置を図 2.10、図 2.11、図 2.12 に示す。また、卍型試験体におけるそれぞれのガラスは図 2.8 に示す番号で呼ぶこととする。

次節以降でまとめる実験結果において、荷重変形曲線の荷重は図 2.9 のロードセルで計測している。また、変形は図 2.10 における CH1 及び CH2 の平均から試験体上部の変位を計測し、CH3 及び CH4 の平均から試験体下部の変位を計測し、その差を試験体の変形としている（式(2.1)）。

$$\text{変形} = \frac{(CH1 + CH2)}{2} - \frac{(CH3 + CH4)}{2} \tag{2.1}$$

表 2.1　卍型試験体諸元

試験体名	H 型鋼製骨組断面形状	ガラス板厚(mm)	錫板厚(mm)
卍型試験体 13	H-15x12x4.5x3	8	1
卍型試験体 14	H-15x16x4.5x3	8	1

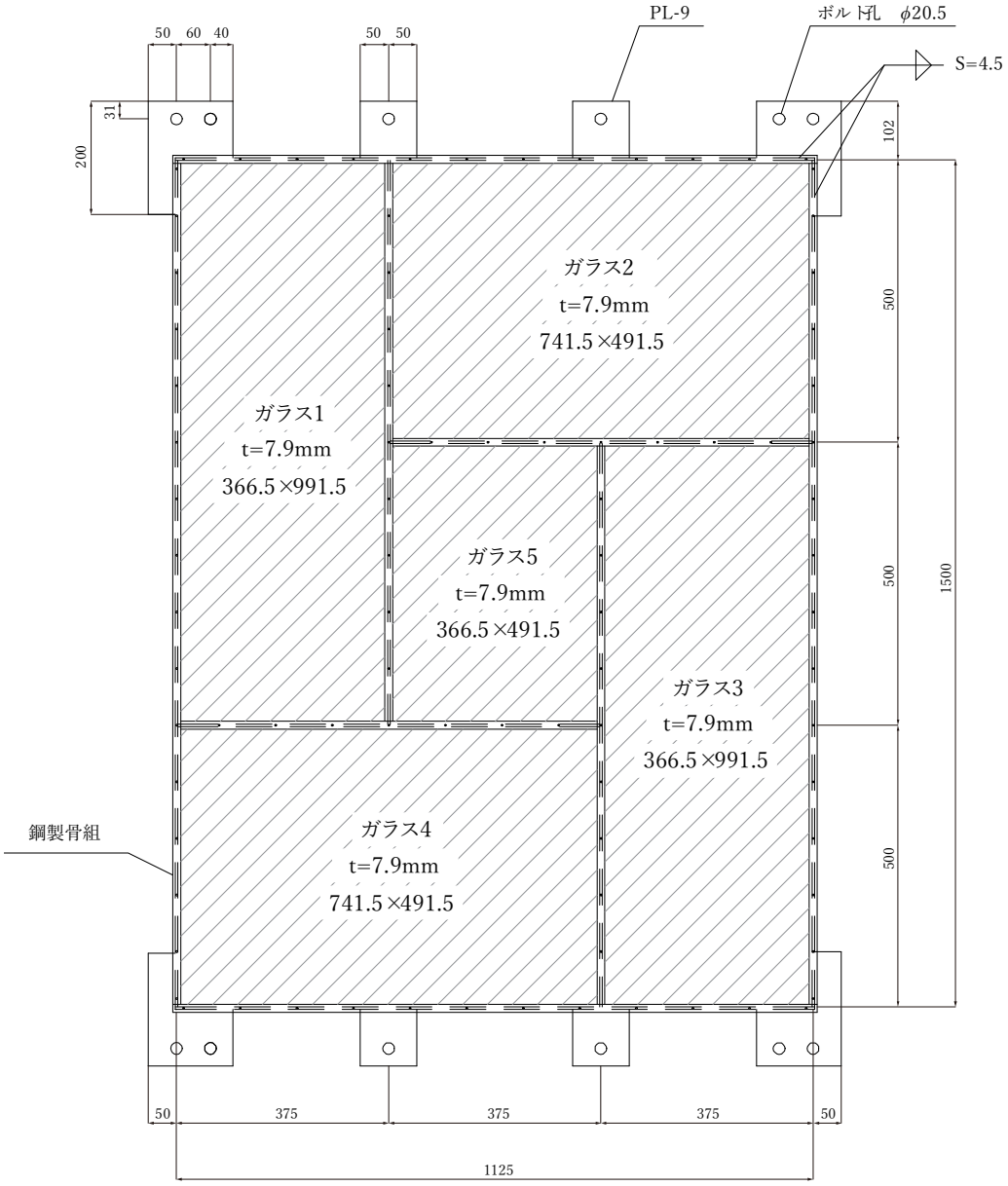

PL-9
ボル ト孔 φ20.5
S=4.5
50 60 40
50 50
31
200
102
500
500
500
1500
ガラス2
t=7.9mm
741.5 ×491.5
ガラス1
t=7.9mm
366.5 ×991.5
ガラス5
t=7.9mm
366.5 ×491.5
ガラス3
t=7.9mm
366.5 ×991.5
ガラス4
t=7.9mm
741.5 ×491.5
鋼製骨組
50
375
375
375
50
1125

図 2.8　卍型試験体寸法

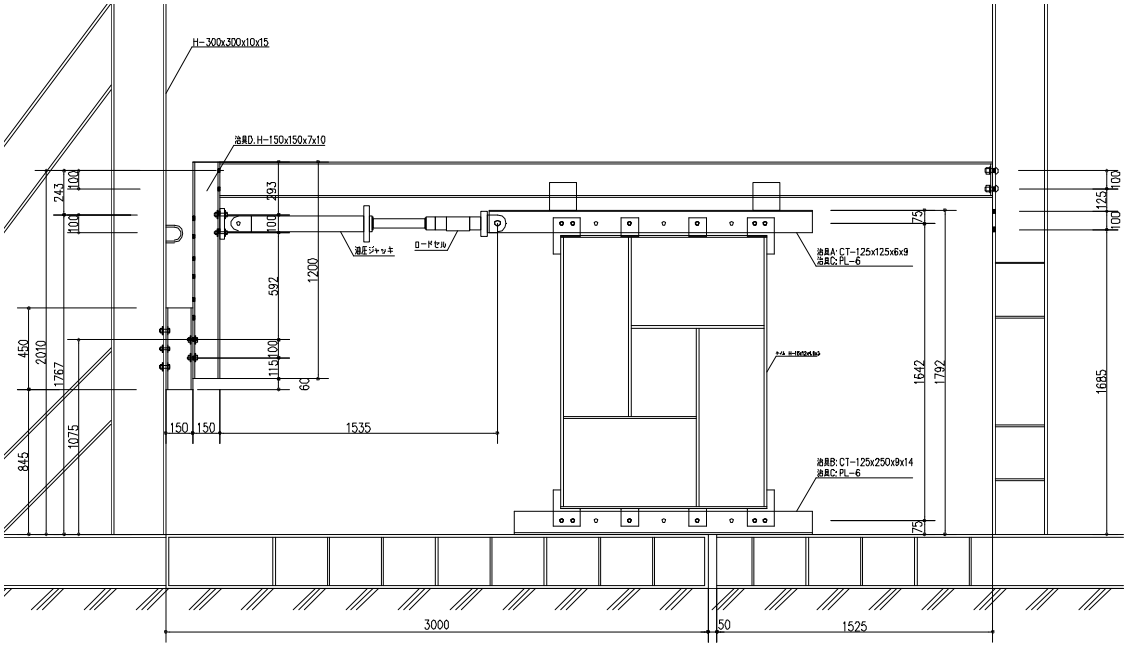


図 2.9　卍型試験体セットアップ図

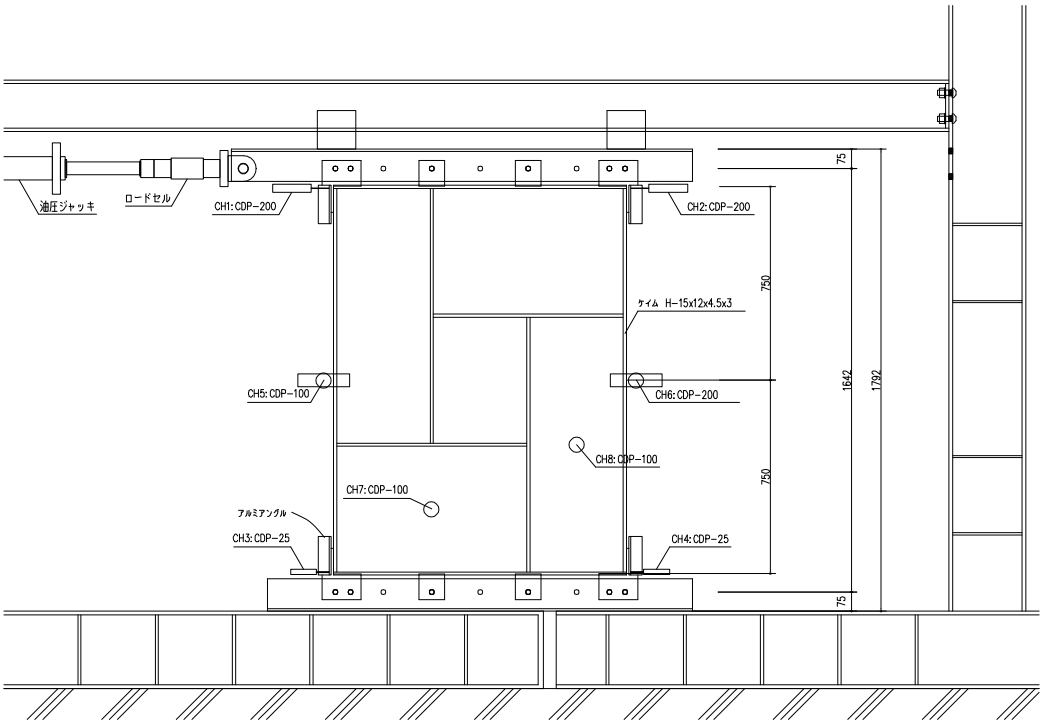


図 2.10　卍型試験体変位計設置位置図

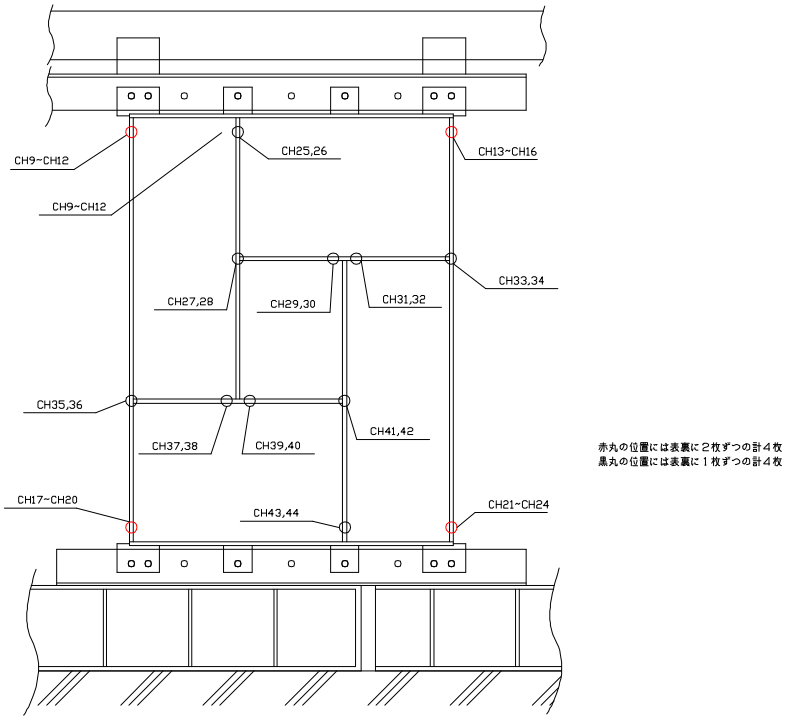


図 2.11　鉄骨ひずみゲージ貼り付け位置

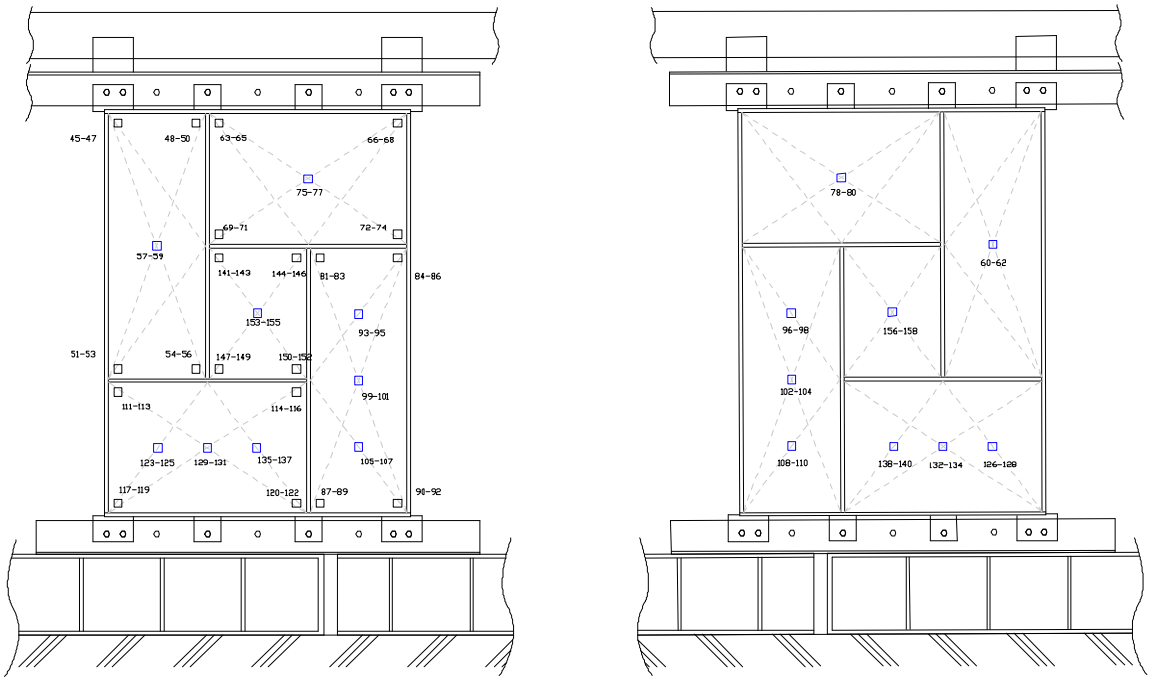


図 2.12　ガラスひずみゲージ貼り付け位置（左…表側、右…裏側）

2.3.2 卍型試験体 13

フランジ幅 12mm の卍型試験体 13 の載荷試験結果を以下にまとめる。

荷重変形曲線（図 2.13）より、繰り返し載荷のサイクル毎に正方向載荷時は+11.5mm 程度のスリップ領域が見られた後、1.00（kN/mm）程度の初期剛性で荷重は上昇する。また、負方向載荷時は-10.5mm 程度のスリップ領域が見られた後、1.06（kN/mm）程度の初期剛性で試験体は剛性を発揮する。

卍型試験体 13 は、変形が+34.1mm で（図 2.14）に示すガラス 3 の下部に最初の亀裂が発生した。その後正方向への載荷は終了し、負方向への載荷を行った。その後の負方向載荷では、変形が-13.7mm の時点でガラス 3 の割れが進行し（図 2.15）、載荷を続けると変形-27.4mm でガラス 3、ガラス 4 に割れが発生し、耐力が失われたため、載荷を終了した（図 2.16、図 2.17）。

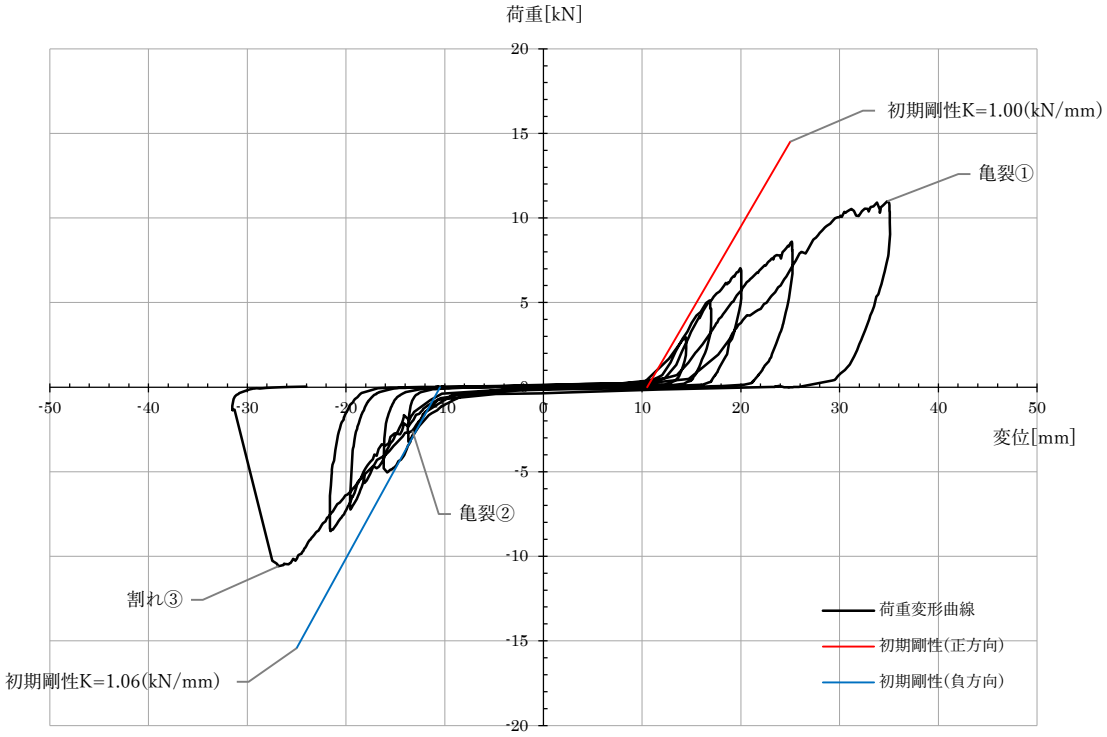


図 2.13　荷重変形曲線（卍型試験体 13）

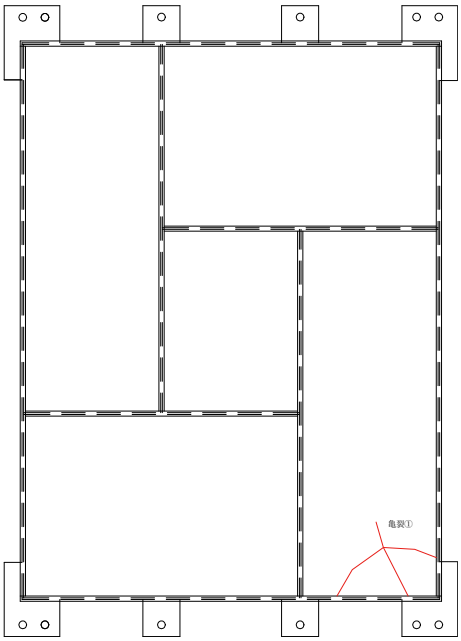


図 2.14　卍型試験体 13 損傷図（1）

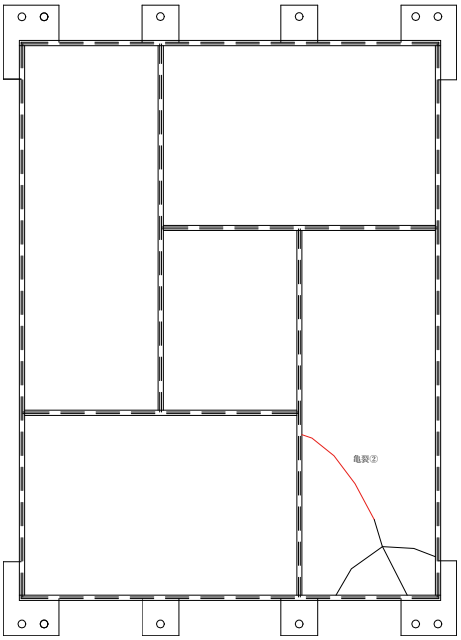


図 2.15　卍型試験体 13 損傷図（2）

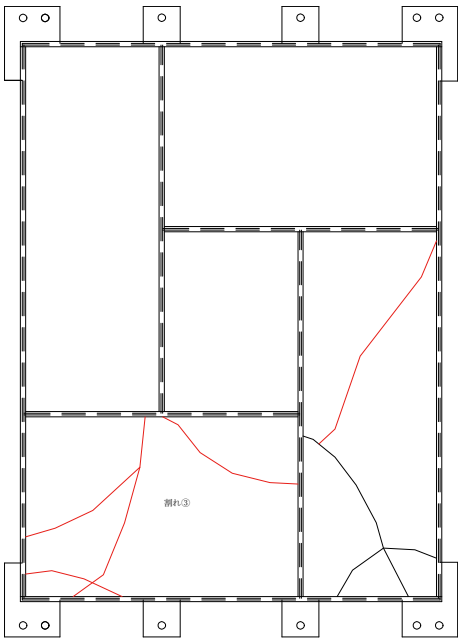


図 2.16　卍型試験体 13 損傷図（3）

図 2.17　載荷終了写真

2.3.3 卍型試験体 14

フランジ幅 16mm の卍型試験体 14 の載荷試験結果を以下にまとめる。

荷重変形曲線（図 2.18）より、繰り返し載荷のサイクル毎に正方向載荷時は+16.0mm 程度のスリップ領域が見られた後、0.90（kN/mm）程度の初期剛性で荷重は上昇する。また、負方向載荷時は-18.0mm 程度のスリップ領域が見られた後、0.90（kN/mm）程度の初期剛性で試験体は剛性を発揮する。

卍型試験体 14 は、変形が+43.5mm となった時点でガラス 2 及びガラス 3 が（図 2.20、図 2.19）のように破壊され、耐力が低下したため、除荷した。その後負方向に載荷を行ったが、剛性は発揮されず、変形が-56.6mm となった時点でガラス 1 が破壊され、耐力が低下したため、載荷試験を終了した（図 2.21、図 2.22）。

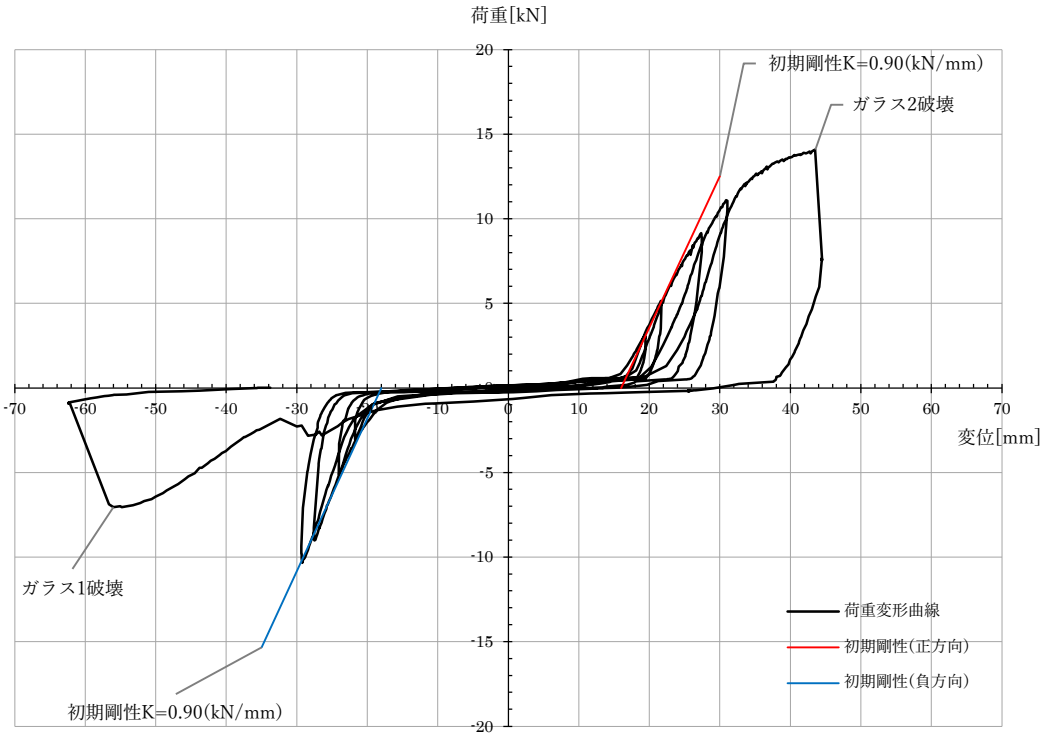


図 2.18　荷重変形曲線（卍型試験体 14）

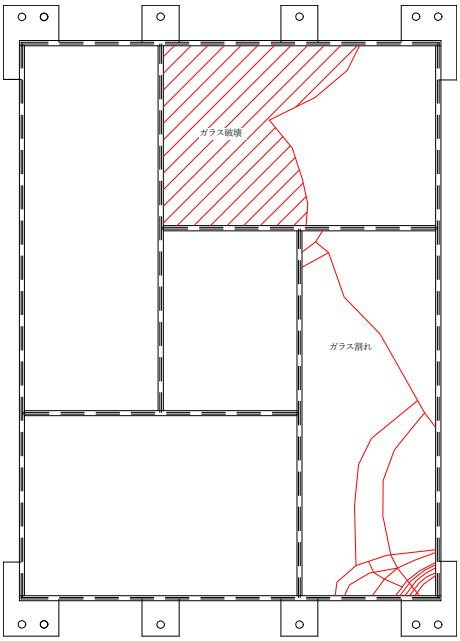


図 2.20　卍型試験体 14 損傷図（1）

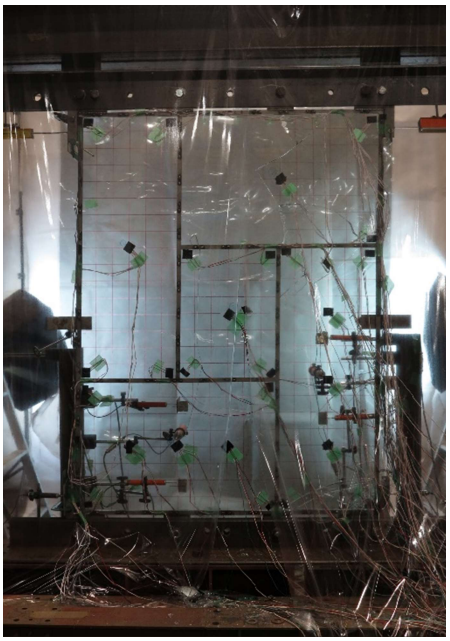


図 2.19　損傷経過写真（1）

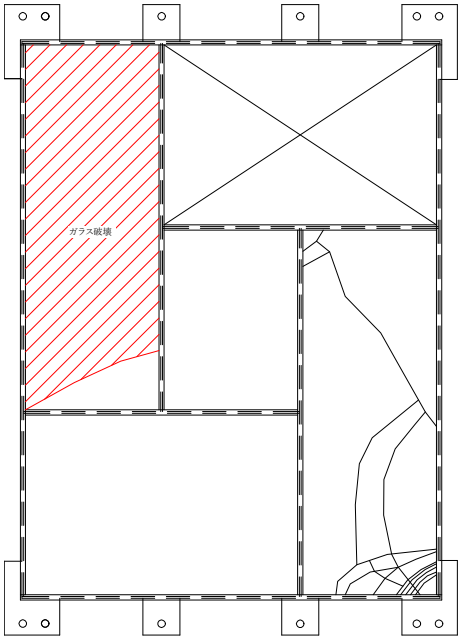


図 2.21　卍型試験体 14 損傷図（2）

図 2.22　損傷経過写真（2）

2.4 小型試験体載荷試験

2.4.1 載荷試験概要

小型試験体の載荷試験は H 型鋼製骨組のフランジ幅が 14mm のものを 3 体行った。試験体の寸法は鋼製骨組の新寸法が[幅×高さ]が[375mm×1000mm]であり、ガラスは厚さが t=7.9mm で 4 隅の角が 5mm 面取りされたフロート板ガラス[366.5mm×991.5mm]を用いている。錫の厚さ、及び載荷方法は各試験体により異なっており、以下にまとめる。

小型試験体 4 は厚さ 1mm の錫が緩衝材として挿入されている。載荷方法は漸増であり、試験体が降伏状態となる手前で載荷を終了し、除荷を行った。また、載荷は正方向 1 回のみ行い載荷試験を終了した。

小型試験体 5 は厚さ 1mm の錫が緩衝材として挿入されている。載荷方法は漸増であり、試験体が降伏状態となる手前までの範囲で繰り返し載荷を行い、載荷試験を終了した。

小型試験体 6 は厚さ 2mm の錫が緩衝材として挿入されている。載荷方法は漸増であり、試験体が降伏状態となる手前までの範囲で繰り返し載荷を行った後、ガラスが破損し耐力がなくなったとみなした時点で除荷し、載荷試験を終了した。

小型試験体の形状は図 2.2、に示す形状である。小型試験体 4 のセットアップ図を図 2.23 に、小型試験体 5 及び小型試験体 6 のセットアップ図を図 2.24 に示す。また、計測器の設置位置および歪ゲージの貼り付け位置を図 2.25、図 2.26、図 2.27 に示す。

次節以降でまとめる実験結果において、荷重変形曲線の荷重は図 2.23、図 2.24 のロードセルで計測している。また、変形は図 2.25 における CH1 及び CH7 の平均から試験体上部の変位を計測し、CH8 及び CH13 の平均から試験体下部の変位を計測し、その差を試験体の変形としている（式(2.2)）。また、3 章の 3.4 節におけるガラスの回転角度は CH18 及び CH19 の 2 点で計測したガラスの面内方向の変位と 2 点間の距離から求める（式(2.3)）。

$$\text{変形(mm)} = \frac{(CH1 + CH7)}{2} - \frac{(CH8 + CH13)}{2} \tag{2.2}$$

$$\text{ガラスの回転角度(rad)} = \tan^{-1}\left(\frac{CH18 - CH19}{300}\right) \tag{2.3}$$

表 2.2　小型試験体諸元

試験体名	H 型鋼製骨組断面形状	ガラス板厚(mm)	錫板厚(mm)
小型試験体 4	H-15x14x4.5x3	8	1
小型試験体 5	H-15x14x4.5x3	8	1
小型試験体 6	H-15x14x4.5x3	8	2

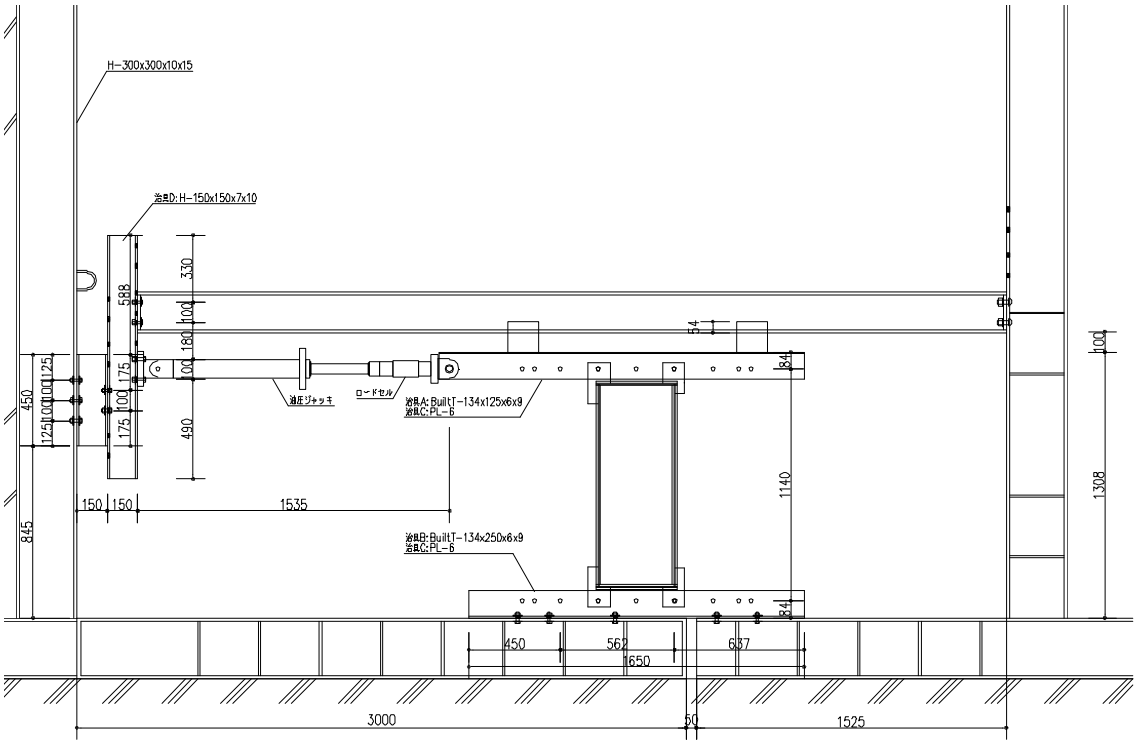


図 2.23　小型試験体セットアップ図(小型試験体 4)

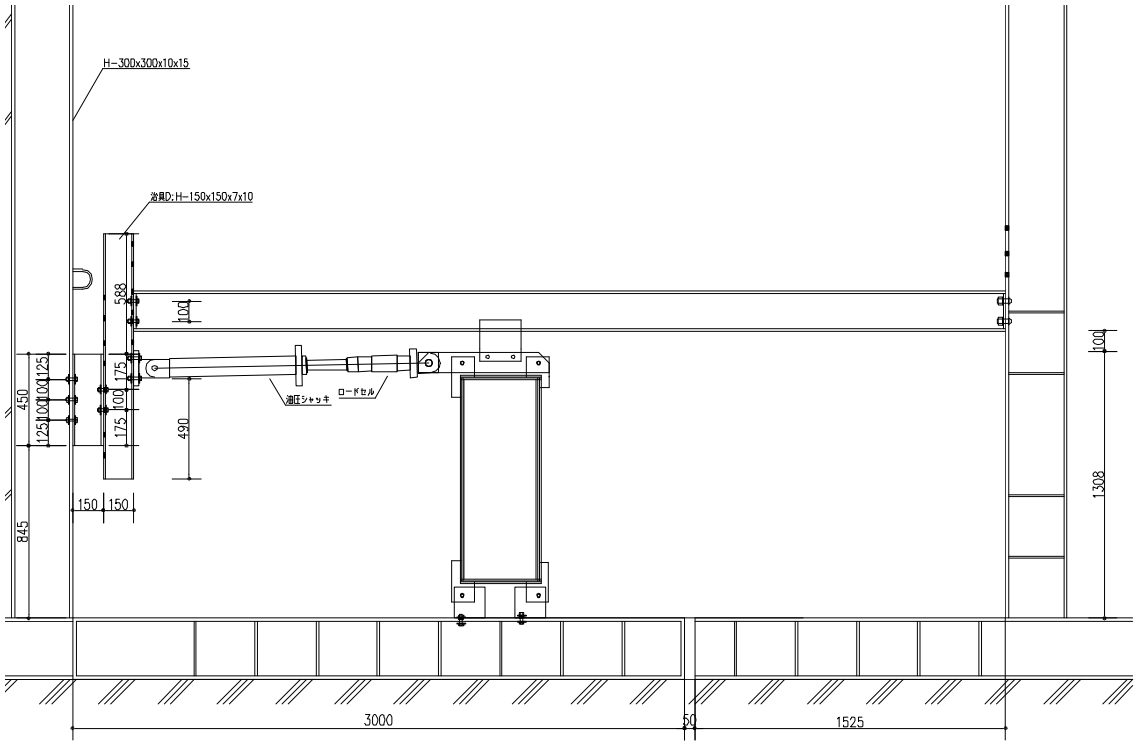


図 2.24　小型試験体セットアップ図(小型試験体 5,6)

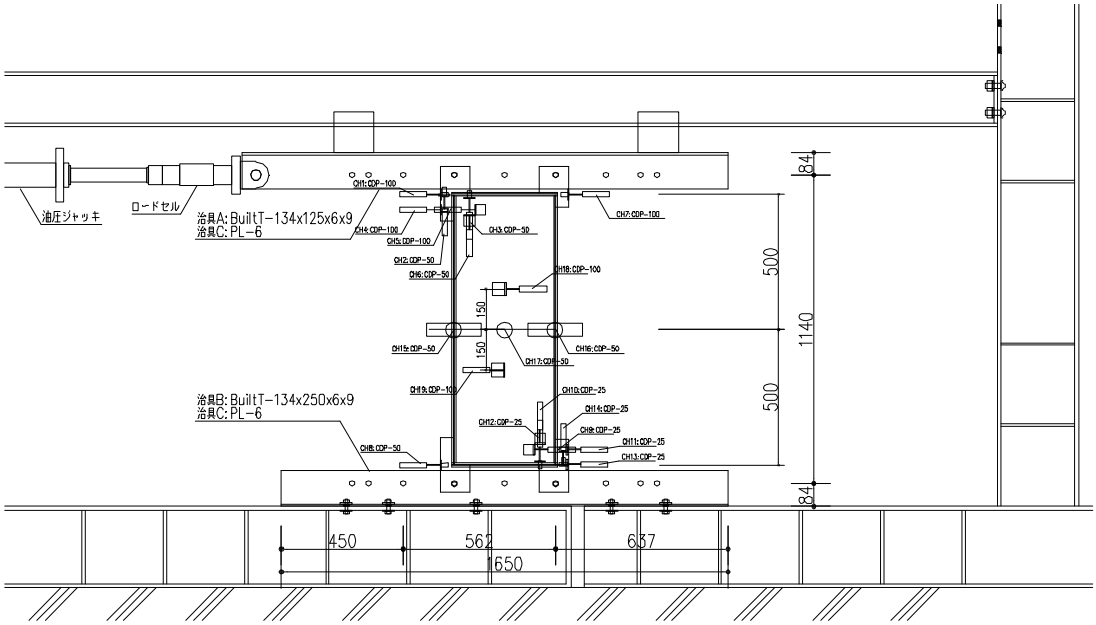
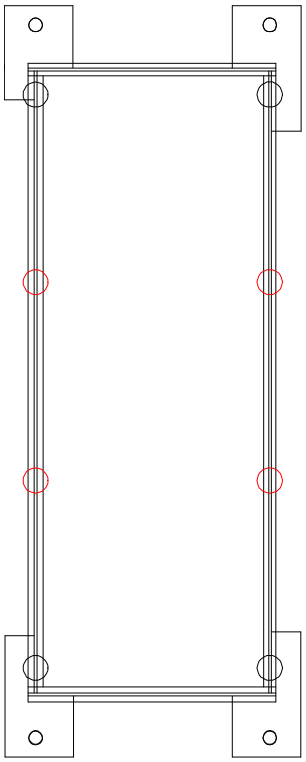
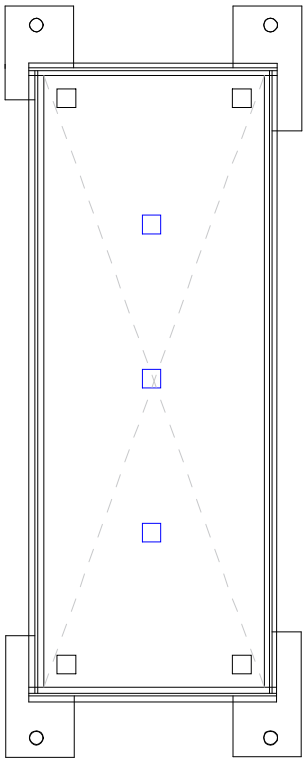


図 2.25　小型試験体変位計設置位置図



黒丸…裏表に 2 枚ずつ
赤丸…裏表に 2 枚とウェブに 1 枚

図 2.26　鉄骨ひずみゲージ貼り付け位置



黒四角…表面に 1 枚
青四角…裏表に 1 枚ずつ

図 2.27　ガラスひずみゲージ貼り付け位置

2.4.2 小型試験体 4

小型試験体 4 の載荷試験結果を以下にまとめる。

荷重変形曲線は図 2.28 のようになり、+16mm 程度のスリップ領域が見られた後、0.581(kN/mm)程度の初期剛性で荷重が上昇した。荷重が 4kN に達した時点で載荷を止め、除荷を行い、載荷試験は終了した。鉄骨及びガラスに損傷した部分はなく、錫の潰れ具合を観察した。左上の角部分では、図 2.30 のように主にガラスの上部と上側の鉄骨の間で錫が圧縮されている様子が観察される。また、右下の角部分では、図 2.30 のように主にガラスの下部と下側の鉄骨の間で錫が圧縮されている様子が分かる。

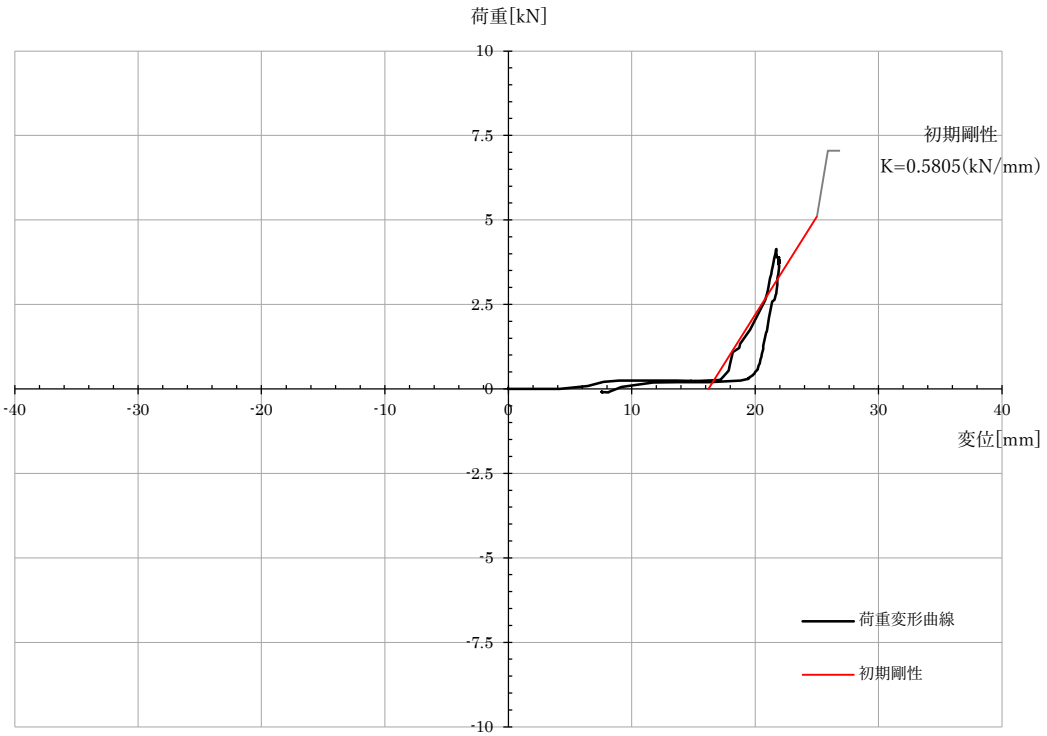


図 2.28　荷重変形曲線（小型試験体 4）

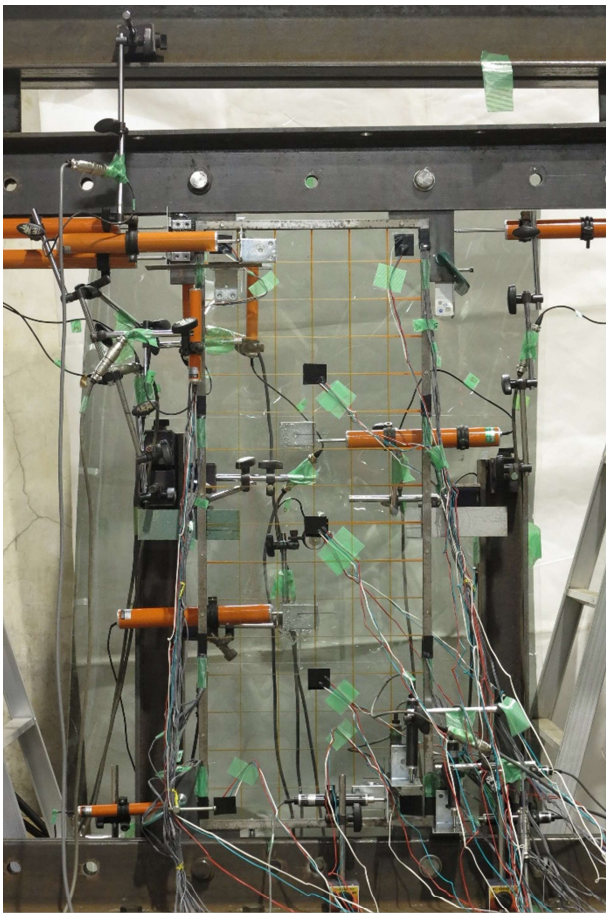


図 2.29　小型試験体 4

図 2.30　実験後の錫の様子（小型試験体 4）

2.4.3 小型試験体 5

小型試験体 5 の載荷試験結果を以下にまとめる。

荷重変形曲線（図 2.31）より繰り返し載荷のサイクル毎に正方向載荷時は+9.5mm 程度のスリップ領域が見られた後、0.538（kN/mm）程度の初期剛性で荷重は上昇する。また、負方向載荷時は-8.0mm 程度のスリップ領域が見られた後、0.534（kN/mm）程度の初期剛性で試験体は剛性を発揮する。

また、正方向に 1 回のみ載荷を行った小型試験体 4 の左上と右下の錫の潰れ具合を比較すると、小型試験体 5 の方がより錫が押しつぶされている様子が分かる（図 2.33）。

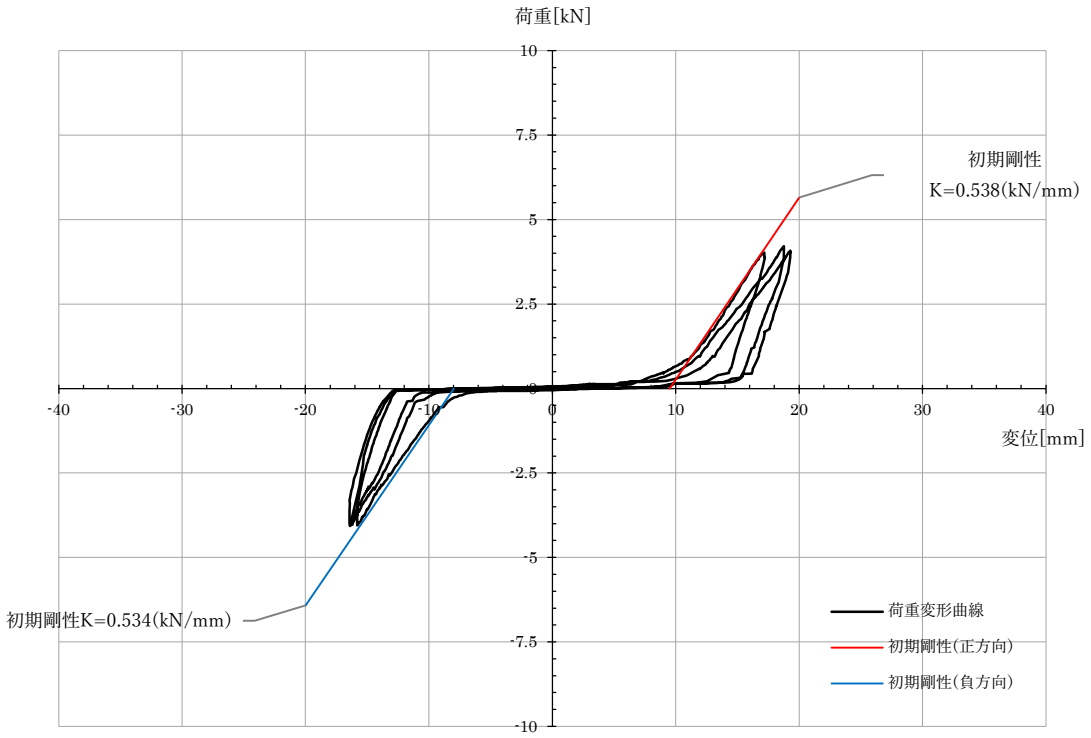


図 2.31　荷重変形曲線（小型試験体 5）

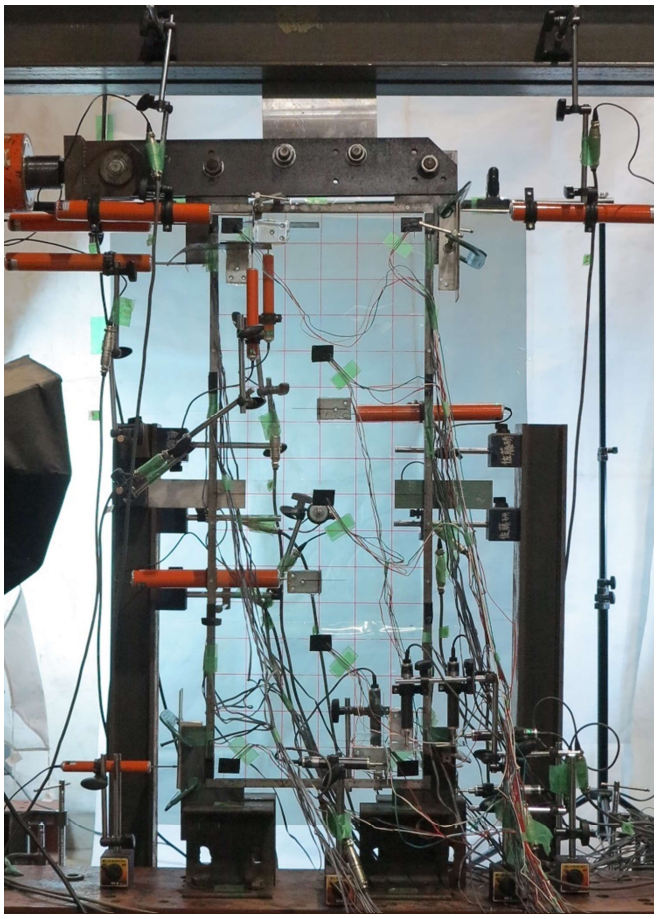


図 2.32　小型試験体 5

図 2.33　実験後の錫の様子（小型試験体 5）

2.4.4 小型試験体 6

小型試験体 6 の載荷試験結果を以下にまとめる。

荷重変形曲線より繰り返し載荷のサイクル毎に正方向載荷時は+2.5mm 程度のスリップ領域が見られた後、0.466（kN/mm）程度の初期剛性で荷重は上昇する。また、負方向載荷時は-3.5mm 程度のスリップ領域が見られた後、0.600（kN/mm）程度の初期剛性で試験体は剛性を発揮する。

また、正方向載荷においては 1 サイクル目、2 サイクル目での試験体の剛性はそれぞれ 0.466（kN/mm）と 0.512（kN/mm）であるのに対し、3 サイクル目、4 サイクル目での剛性は 0.697（kN/mm）と 0.707（kN/mm）と上昇している。この相違は、1、2 サイクル目では錫は弾性範囲で挙動しているが、3、4 サイクル目では錫は十分に押しつぶされ、塑性範囲で挙動していることによるものではないかと考えられる。

小型試験体 6 は変形が+30.3mm となった時点で、に示す箇所で鉄骨が破断し、耐力が低下した。また、小型試験体の左側の鉄骨において、ウェブのビス穴があけられている箇所が引き延ばされており、試験体の左上と左下の治具が溶接された範囲のビスに変形、破断が生じていた。

この載荷試験より、緩衝材として挿入する錫の厚さが 1mm の場合と 2mm の場合とでは初期剛性の値に大きな影響はなく、材料使用量および、コストの面から錫緩衝材の厚さは 1mm が良いことが分かった。

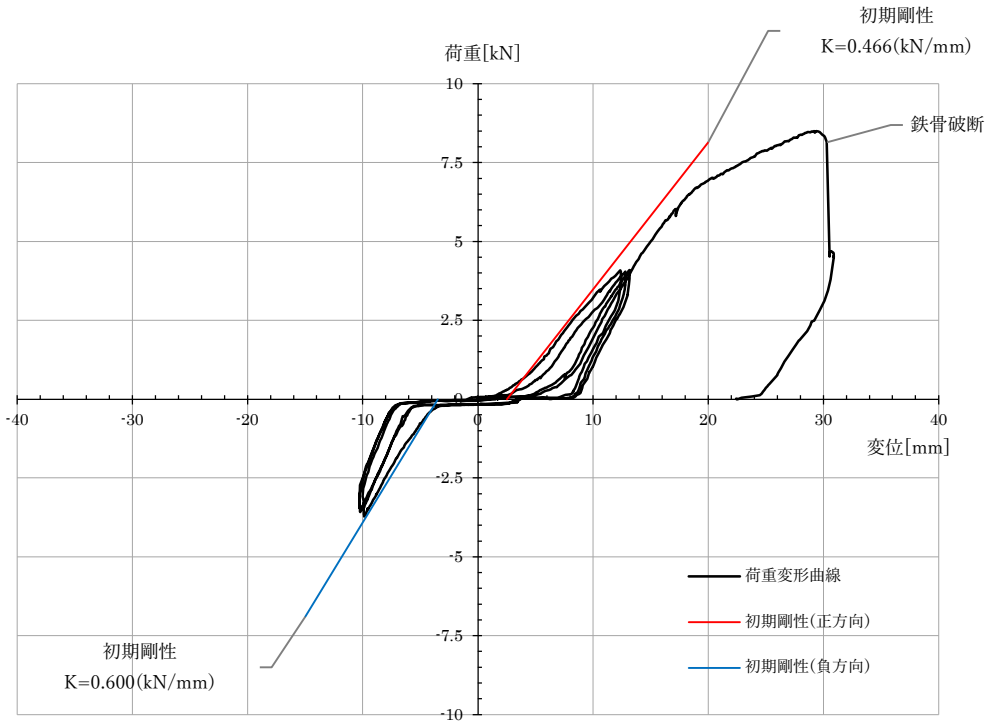


図 2.34　荷重変形曲線（小型試験体 6）

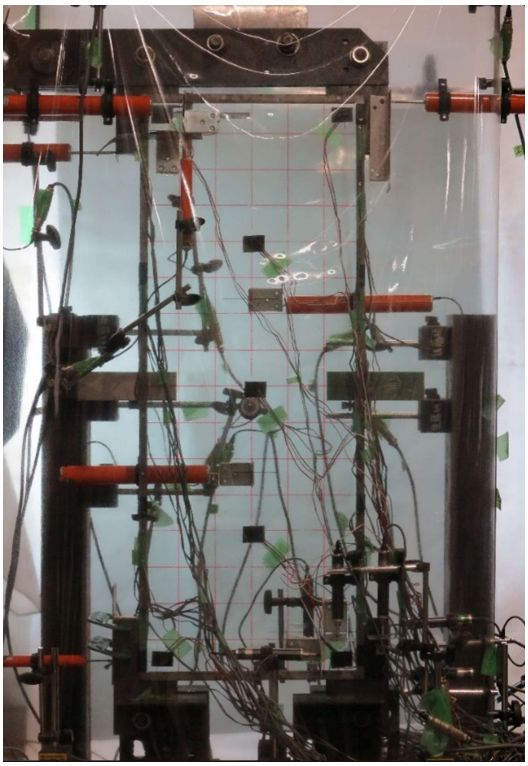


図 2.35　小型試験体 6

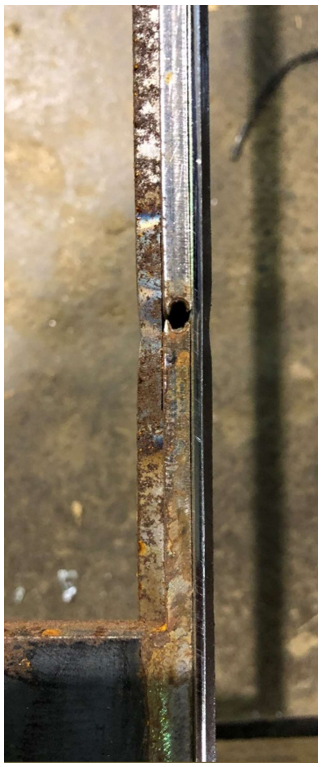


図 2.36　鉄骨破断の様子

図 2.37　ビスが変形、破断している様子

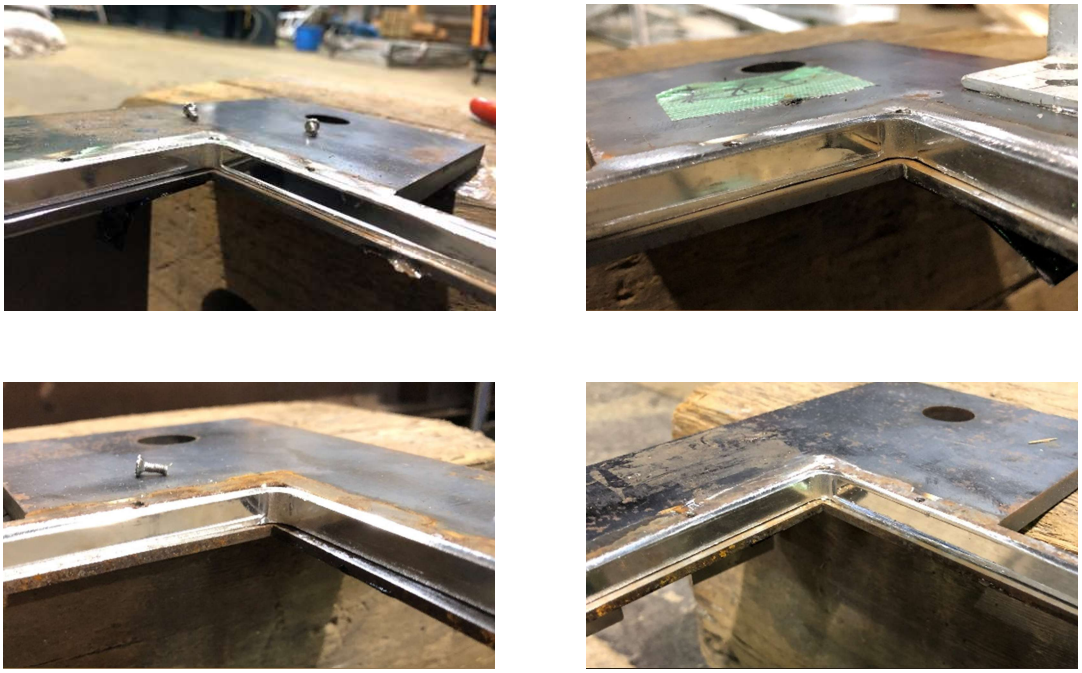


図 2.38　実験後の錫の様子（小型試験体 6）

第3章 モデル化の手法

3.1 概説

本章では、試験体における各部材の剛性の評価方法及びモデル化の手法について述べる。

3.2 節では、鉄骨を 1 本の部材ではなくより細かい部材に分けることで、作成される試験体の形状をより正確に解析モデルに反映するモデル化について述べ、解析モデルに入力する骨組材の諸元を示す。

3.3 節では、既往研究[8]で行われた錫をガラスの角で圧縮する錫圧縮試験の結果をもとにした錫の材料非線形性のモデル化について述べ、解析モデルで用いる錫ばね材の諸元を示す。

3.4 節では、ガラスが鉄骨で拘束される位置のモデル化について述べる。

3.5 節では、既往研究[12]において提案されたガラスと錫の隙間のモデル化の手法について述べる。

3.2 鉄骨の細分化

ステンドグラス構造体に用いられている鋼製骨組は、ウェブとフランジを別々で生成し、溶接及びビス止めにより H 形鋼状に組み立てられている。卍型試験体においてフランジ材は 1 枚の鋼板から卍型の形状に作成されており一体の部材だが、ウェブ材は図 3.1 の赤丸に位置において写真のように溶接で接合されていない部分があり一体の部材ではない。また、載荷試験終了後には変形が見られるため、試験体の剛性に影響を与えている可能性がある。したがって、本研究では鉄骨を細分化したモデルを作成し、初期剛性への影響を検証する。

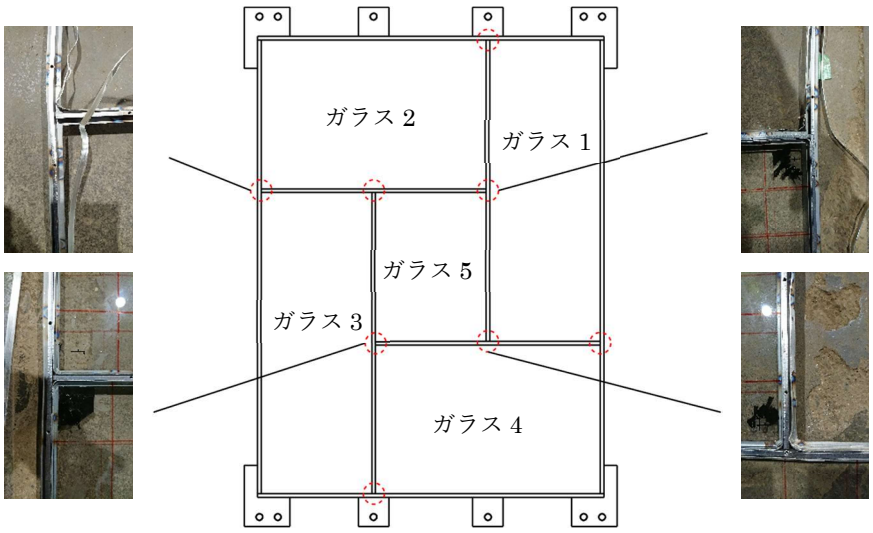


図 3.1 ウェブ材が分離している位置

鉄骨は図 3.2 のようにウェブ材、フランジ材、ビス材、溶接材に細分化してモデル化し、卍型試験体において分離していた部分にはウェブ材を設けないこととした。

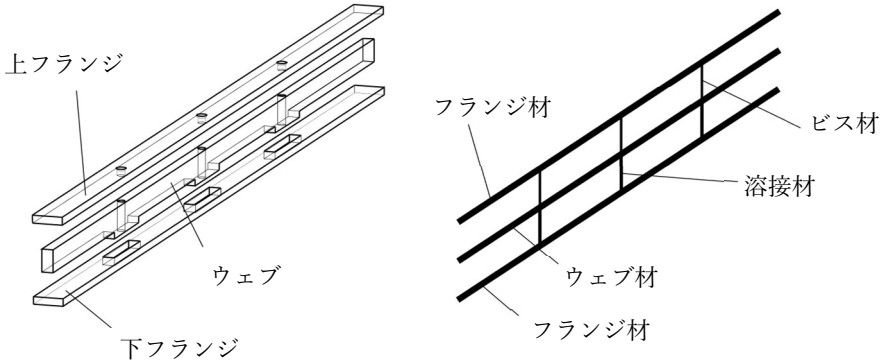


図 3.2 鉄骨の細分化によるモデル化

3.3 錫の材料非線形性のモデル化

錫の剛性は既往研究 [8]で行われた錫の圧縮試験から評価する（図 3.3)。錫は極めて柔らかい金属であり、ガラスから圧縮力を受けるとのように次第にかたくなり、変形しにくくなるという特徴がみられる（図 3.4)。したがって、圧縮力を受けた錫ばねの縮み Δ_{tin} に応じて錫ばね部材の剛性を変化させることで錫の材料非線形性をモデル化する。既往研究 [8]で行われた錫圧縮試験では図 3.5 のようにガラスの角で錫が押しつぶされ、錫が縮むと錫がガラスによって押しつぶされている範囲も増加する。そこで、本研究では錫圧縮試験の試験体 7 の実験結果を用いて図 3.5 の錫の押しつぶされた面積と荷重の関係を用いて錫の剛性を評価する。

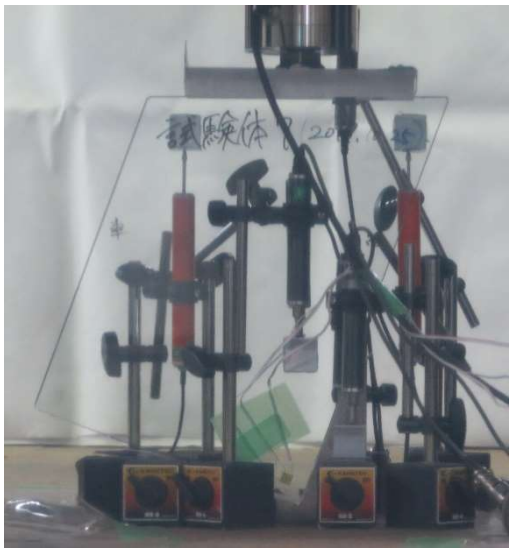


図 3.3　錫圧縮試験

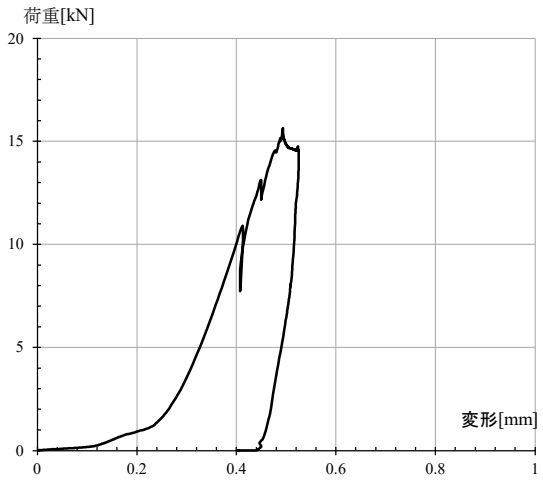


図 3.4　錫圧縮試験の荷重変形曲線

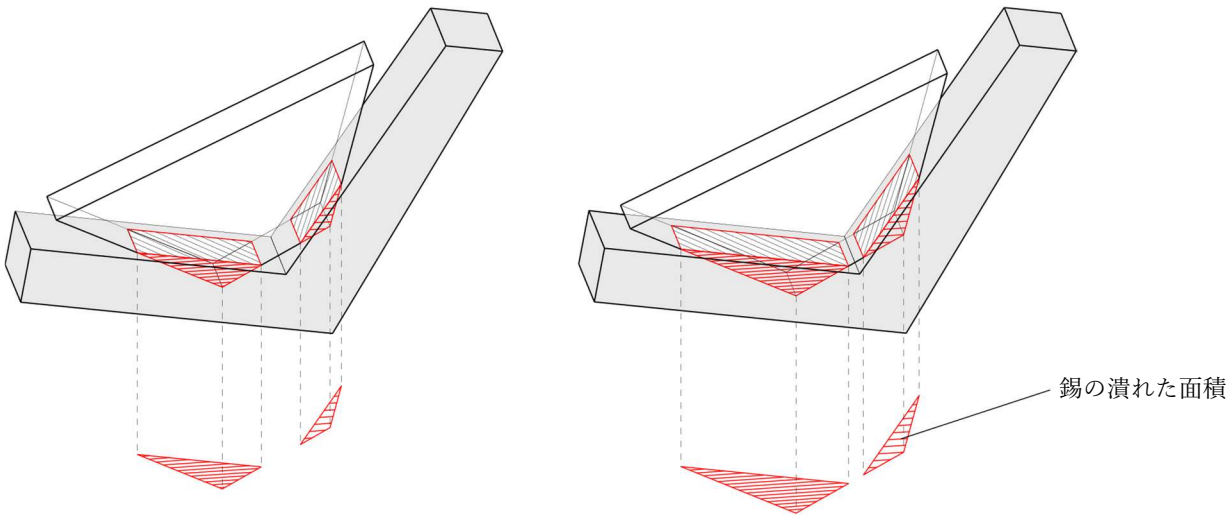


図 3.5 錫の圧縮される様子と錫の潰れた面積

錫の押しつぶされた面積と荷重の関係（図 3.6）から、錫ばねのばね定数を荷重が 0~1kN（$0 < \Delta_{tin} \leqq 0.0862$mm）の範囲では 1.018（kN/mm）、1~2.5kN（$0.0862 < \Delta_{tin} \leqq 0.1126$mm）の範囲では 2.175（kN/mm）、2.5kN 以上（$0.1126\text{mm} < \Delta_{tin}$）の範囲では 4.167（kN/mm）と評価した。解析モデルにおいては、式(3.1)、式(3.2)より式(3.3)が成り立つので、部材の断面積の値を変化させることで対応する錫ばねの剛性を入力する。

$$F = k \cdot x \tag{3.1}$$

$$F = \frac{E \cdot A}{L} \cdot x \tag{3.2}$$

$$k = \frac{E \cdot A}{L} \tag{3.3}$$

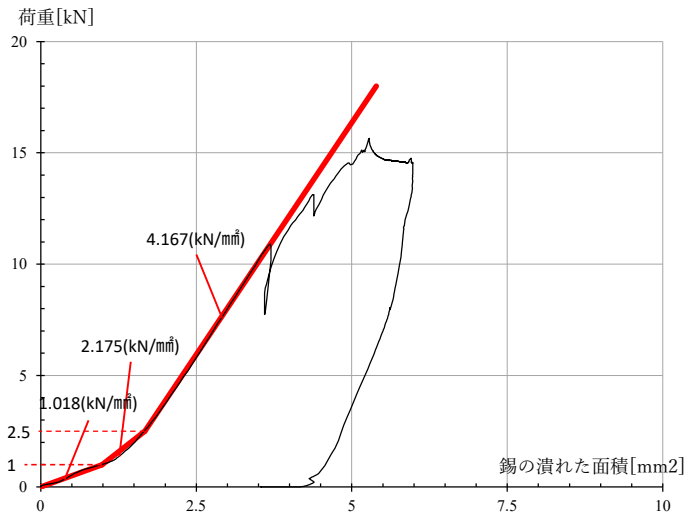


図 3.6　錫の潰れた面積と荷重の関係

表 3.1　縮みに応じて用いる錫のばね定数

錫ばねの縮み（mm）	$0 < \Delta_{tin} \leqq 0.0862$	$0.0862 < \Delta_{tin} \leqq 0.1126$	$0.1126\text{mm} < \Delta_{tin}$
錫のばね定数（kN/㎟）	1.018	2.175	4.167

3.4 ガラス拘束位置のモデル化

まず、ステンドグラス構造体におけるガラスと錫の隙間について小型試験体を用いて述べる。小型試験体の鋼製骨組の大きさは芯々で幅 375mm、高さ 100mm、ウェブ厚は 4.5mm であり、ガラスが拘束される枠としての大きさは幅 370.5mm、高さ 995.5mm である。その鋼製骨組のウェブの内側の枠の中に幅 366mm、高さ 991mm のガラスが拘束され、そのガラスと鋼製骨組の隙間 2.25mm に 1mm の錫が挿入されている（図 3.7)。錫は非常に柔らかく剛性を発揮するのはガラスと鋼製骨組間の距離が 1mm となってからであるため、鋼製骨組の内側に接しているものと考えるとガラスと錫の間には 1.25mm のクリアランスが存在している。

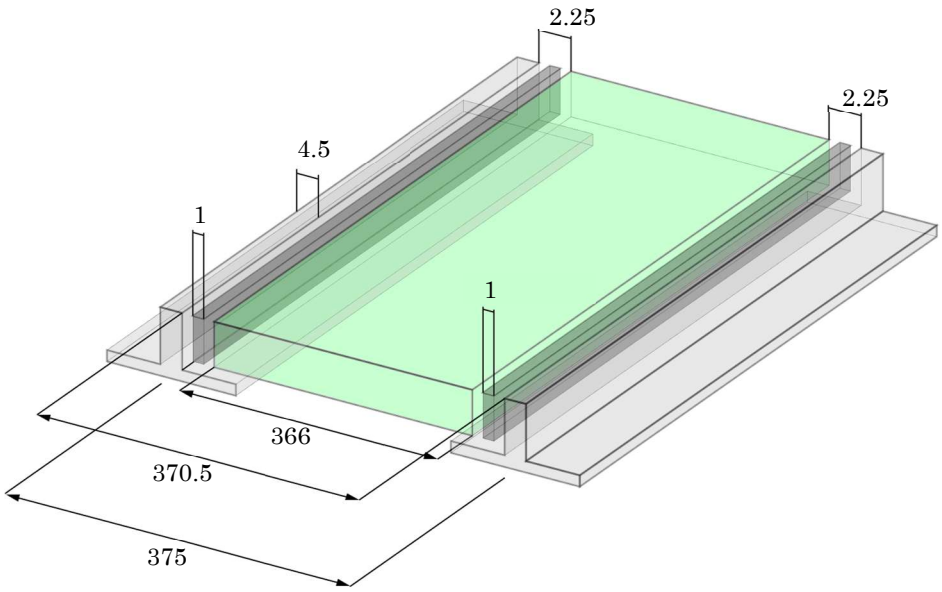


図 3.7 骨組とガラスと錫の位置関係

既往研究[7),8)]における詳細解析モデルでは鋼製骨組を芯々の幅 375mm、高さ 1000mm と同じ大きさの骨組材にモデル化し、その内側に長さ 1mm の錫ばね材をかいしてガラスは幅 373mm、高さ 998mm の大きさとしてモデル化されており、解析におけるガラスの大きさは実際のガラスより大きくモデル化されている（図 3.8)。ステンドグラス構造体はガラスがブレース材として働き、その剛性を発揮することで変形に耐えるという機構であるため、ガラスと錫の隙間を無視し、ガラスを実際より大きくモデル化してしまうことは初期剛性の大きさに影響を与えていると考えられる。したがって、本研究ではガラスが鋼製骨組に拘束されて剛性を発揮し始める位置を解析モデルに反映させることで、ガラスの大きさを再現し、ガラスの拘束位置のモデル化を行う。

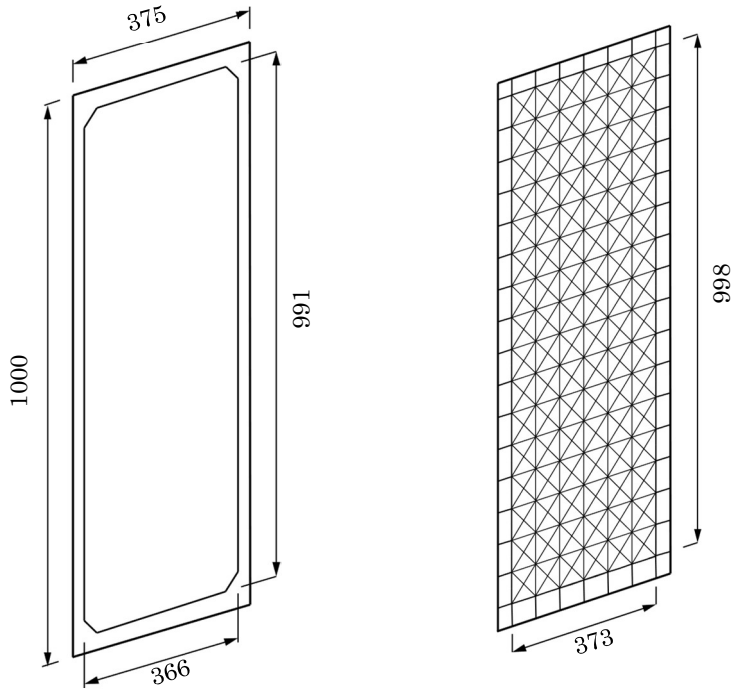


図 3.8 ガラスの大きさの比較

小型試験体 5 においてガラスの回転角度と荷重の関係は図 3.9 のようになり、ガラスは回転した後に骨組に拘束され、剛性を発揮している。また、載荷試験終了後の錫において、正方向載荷時はガラスは左上の角の上側と右下の角の下側で拘束され錫を押しつぶしている様子が、同様に負方向載荷時に関しても右上の角の上側と左下の角の下側の錫が押しつぶされている様子が観察される（図 3.10）。以後試験体は左右対称であるため、正方向載荷時において分析を行う。実験結果から、ガラスがブレース材として拘束される位置は、左上の角の上側の面取りがされた位置から右下の角の下側の面取りがされた位置である（図 3.11）。この 2 点の距離を拘束距離と呼び、拘束距離を用いてガラスの大きさを評価することとする。実験における拘束距離は 1053.0mm である。

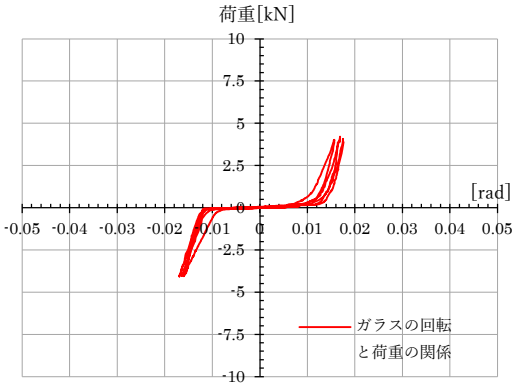


図 3.9 ガラスの回転と荷重の関係

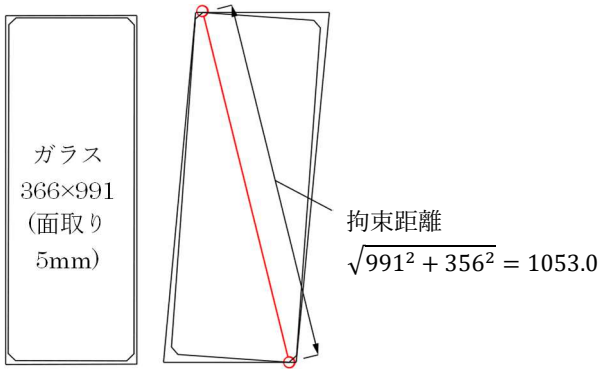


図 3.11 ガラスの拘束位置と拘束距離

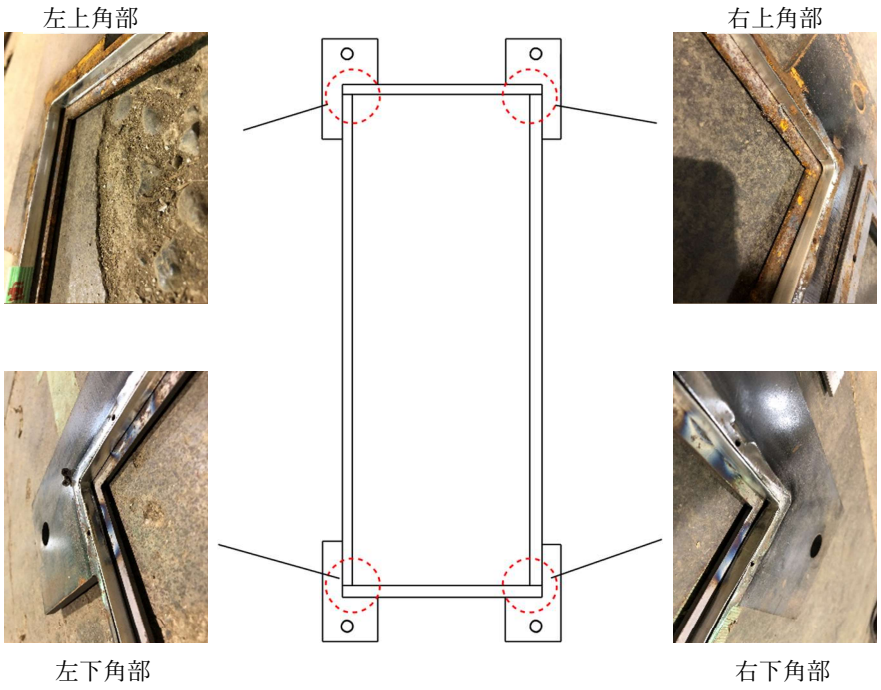


図 3.10 試験体の角部分で錫が圧縮された様子

詳細解析モデルにおいてガラスが骨組に拘束される位置は、ガラス材と骨組材の間に設ける錫ばね材の最端の位置により定まる。したがって、詳細解析モデルにおける拘束距離は最端の錫ばね材の位置の距離により求まる。既往研究[8]におけるメッシュ幅 31.25mm の詳細解析モデル(Model1)はガラスの角部分にも錫ばね材が設けられた解析モデルであり、拘束距離は 1068.0mm である。実験における拘束距離と比較すると乖離がある。Model1 のガラスの角部分の錫ばね材を取り除いた解析モデル(Model2)の拘束距離は 1047.7mm であり、ガラスを小さく評価しすぎてしまう可能性がある。(以後ガラスの角部分の錫ばね材を取り除くことを面取りと呼ぶ。)メッシュ幅を 25mm とした詳細解析モデルにおいて面取りを施した解析モデル(Model3)の拘束距離は 1051.5mm である。

以上から、メッシュ幅 25mm の詳細解析モデルにおいて面取りを施すことにより、ガラスの拘束位置および拘束されるガラスのブレース材としての長さを解析に反映し、ガラスの拘束位置のモデル化を行う。

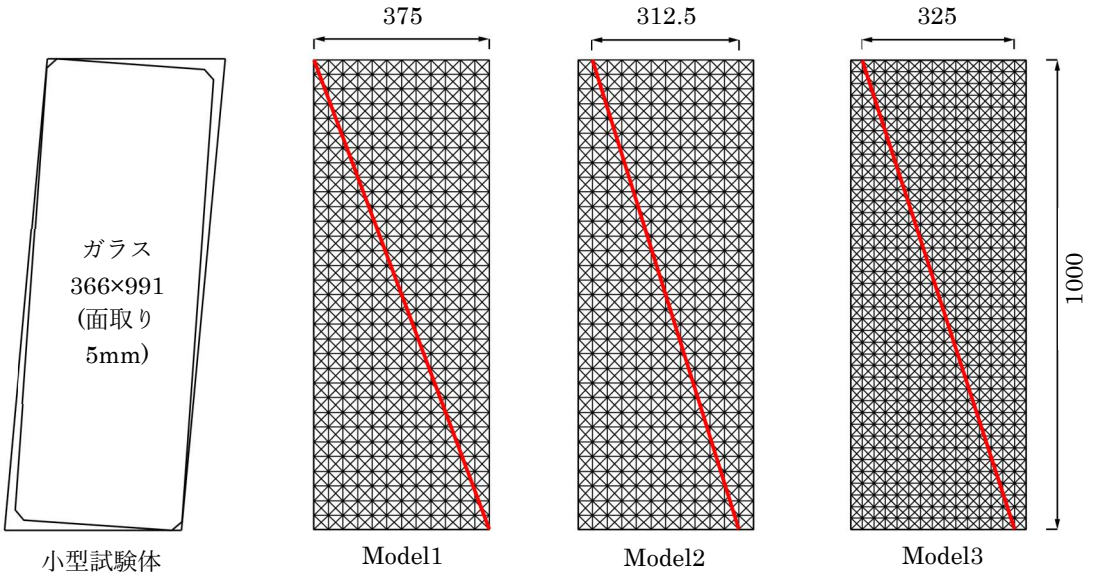


図 3.12 拘束距離の比較

表 3.2 拘束距離まとめ

	小型試験体	Model1	Model2	Model3
拘束距離(mm)	1053	1068	1047.7	1051.5

第4章 錫ばね除去法による解析

4.1 概説

本章では、錫ばね除去法による解析を行い、モデル化手法の有効性を確認する。

4.2 節では、錫ばね除去法による解析の手法について説明する。

4.3 節では、小型試験体の詳細解析モデルを作成し、錫ばね除去法による解析を行った結果についてまとめる。

4.4 節では、卍型試験体において同様に詳細解析モデルを作成し、錫ばね除去捜査による解析及び結果について述べる。

4.5 節では、本章の結論として錫ばね除去法により得られた知見及び課題についてまとめる。

4.2 錫ばね除去法

錫ばね除去による解析の手法について述べる。

詳細解析モデルは図 4.1 のように骨組材とガラス材の隙間に錫ばね材を設けた解析モデルであり、線形解析を行うと引張の働く錫ばね材が発生する。その引張が入った錫ばね材を取り除き新しい解析モデルを作成する。新しい解析モデルにおいて、錫の材料非線形性を解析に反映させるために、錫ばね材に入力する剛性は、1 つ前の解析で錫ばね材に働いた応力度をもとに変化させる。作成された新しい解析モデルで再度線形解析を行う。この操作を引張の働く錫ばね材がなくなり圧縮の働く錫ばね材のみとなるまで繰り返す。既往研究では錫の剛性判定は行われていなかったため、錫ばね材がなくなった時点で解析を終了し、解析モデルにおいて与えた荷重と頂部 2 点の変位の平均から初期剛性を求めていた。しかし、錫の剛性判定を組み込んだ本研究の場合では、引張の入る錫ばね材がなくなった後も錫ばね材に入力する剛性が変化し、初期剛性の数値が変動することや、再び引張の入る錫ばね材が現れることがある。したがって、引張の入る錫ばね材がなくなった後も解析を繰り返し、解析モデルが安定して同じ初期剛性の値を返すようになった時点で解析を終了することとした。また、錫ばね材の剛性判定及び錫ばね除去操作は付録に示す錫ばね除去プログラムを用いて行った。

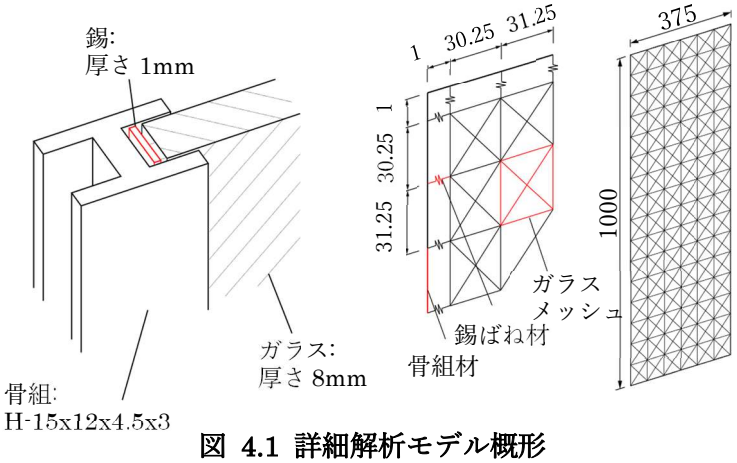


図 4.1 詳細解析モデル概形

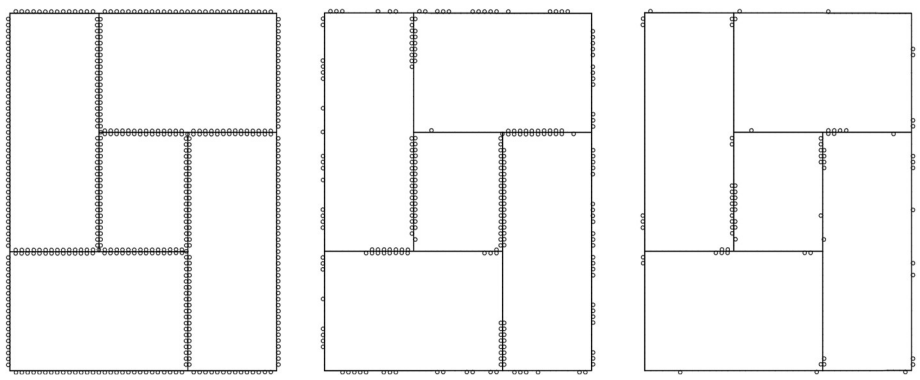


図 4.2 錫ばね材が取り除かれていく様子

4.3 小型試験体の解析

4.3.1 解析モデル種類

3 章で提案したモデル化の手法を用いて小型試験体の詳細解析モデルを作成し、解析を行う。各モデル化の手法がどの程度初期剛性に影響を与えているかを確認するためにモデル化の手法を組み合わせて作成したモデルのモデル化の条件について述べ、表 4.1 にまとめる。

- Model101…試験体を骨組材、ガラス材、錫ばね材に細分化したメッシュ幅 25mm の詳細解析モデルに錫の材料非線形性を反映させたもの。面取り及び鉄骨細分化は行わない。

- Model102…Model101 に面取りを施した詳細解析モデル。

- Model103…Model101 に面取り及び鉄骨の細分化を施した詳細解析モデル。

表 4.1 小型試験体解析モデルの種類及びモデル化条件

	メッシュ幅 (mm)	錫の剛性変化	面取り	鉄骨細分化
Model101	25	あり	なし	なし
Model102	25	あり	あり	なし
Model103	25	あり	あり	あり

4.3.2 解析モデル形状

Model101 の形状は試験体を 25mm 間隔のメッシュ状に細分化した図 4.3 に示す形状であり、鉄骨は H 型の骨組材 1 本でモデル化し、錫ばねはガラスの角部分まで設けられている。ガラスの角部分の錫ばね(錫ばね_端)は角でない間の錫ばねの半分の範囲の錫をモデル化した部材であるため、錫ばね材_間の半分の剛性を与えている。また、ガラスの端の格子材(格子材_端)も格子材_間の半分幅の領域をモデル化した部材であるので、格子材_間の半分の負担幅として剛性を入力している。荷重は実験と同じく頂部 2 点に加え、頂部 2 点の節点条件は x 方向の変位、z 方向の変位、y 軸回りの回転はフリーとし、y 方向の変位、x 軸回りの回転、z 軸回りの回転は固定とした。また、実験において脚部は 2 点で拘束されており、解析での節点条件は y 軸回りの回転のみをフリーとし、残りは固定とした。荷重の加える位置及び節点条件は全てのモデルにおいて同様であるため以降のモデルでは説明を省略する。

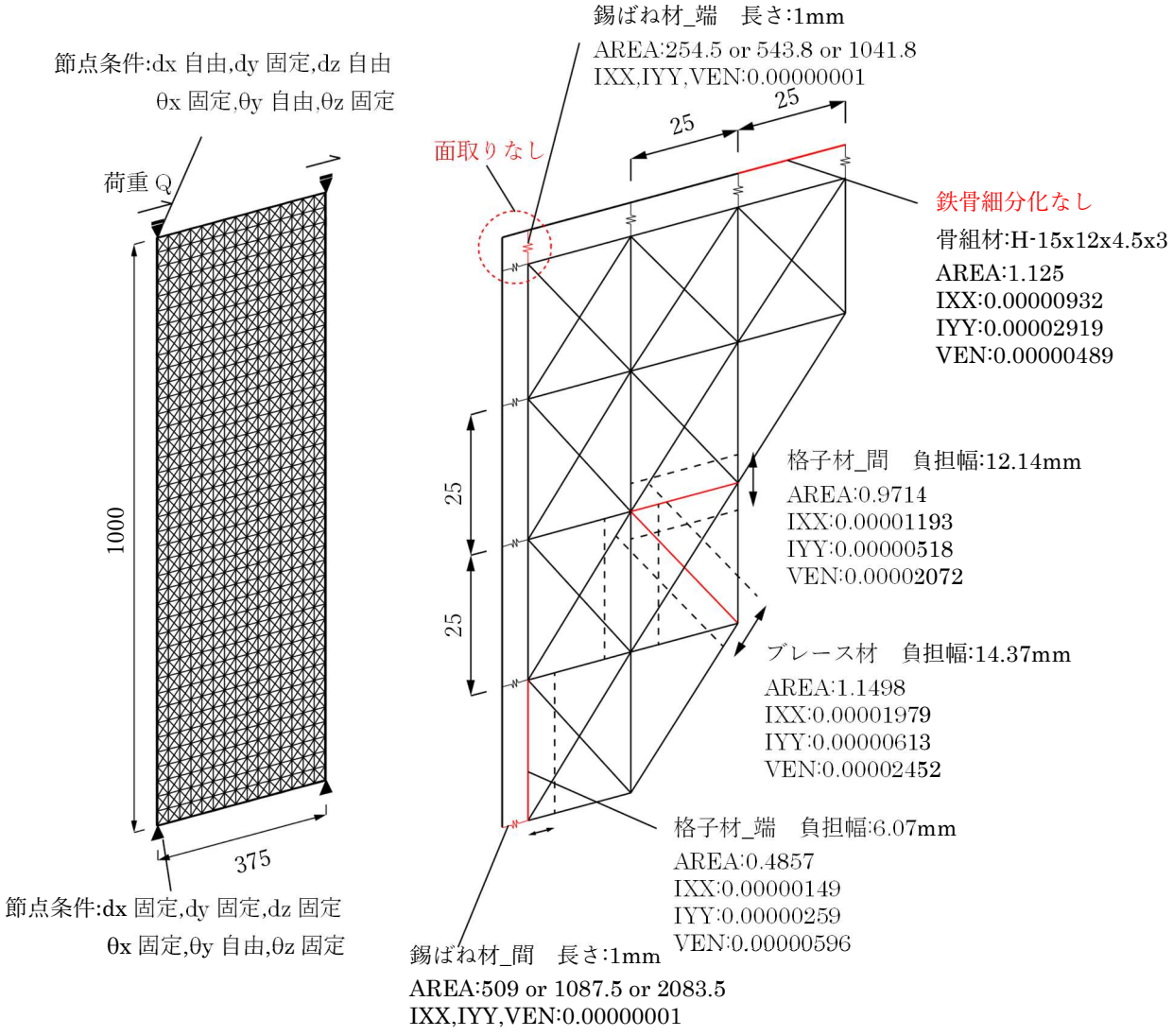


図 4.3 Model101 形状

Model102 は図 4.4 に示す形状であり、錫ばね材をガラスの角部分には設けず面取りが施されている。鉄骨は Model101 と同じく 1 本の骨組材でモデル化している。

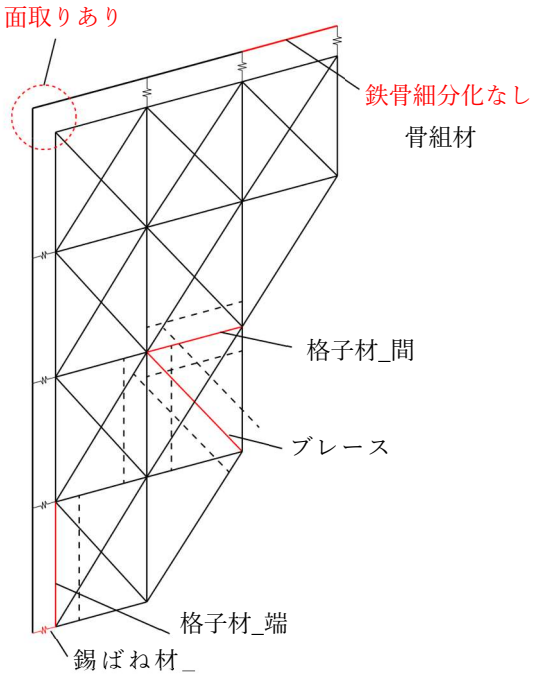


図 4.4 Model102 形状

Model103 は図 4.5 に示す形状であり、ガラスの角部分には錫ばねは設けず、面取りを施し、鉄骨はフランジ材、ウェブ材、ビス材、溶接材に細分化してモデル化されている。試験体の骨組の 4 隅ではウェブとフランジが一体となっているのでビス材のない部分にも溶接材を設けている。

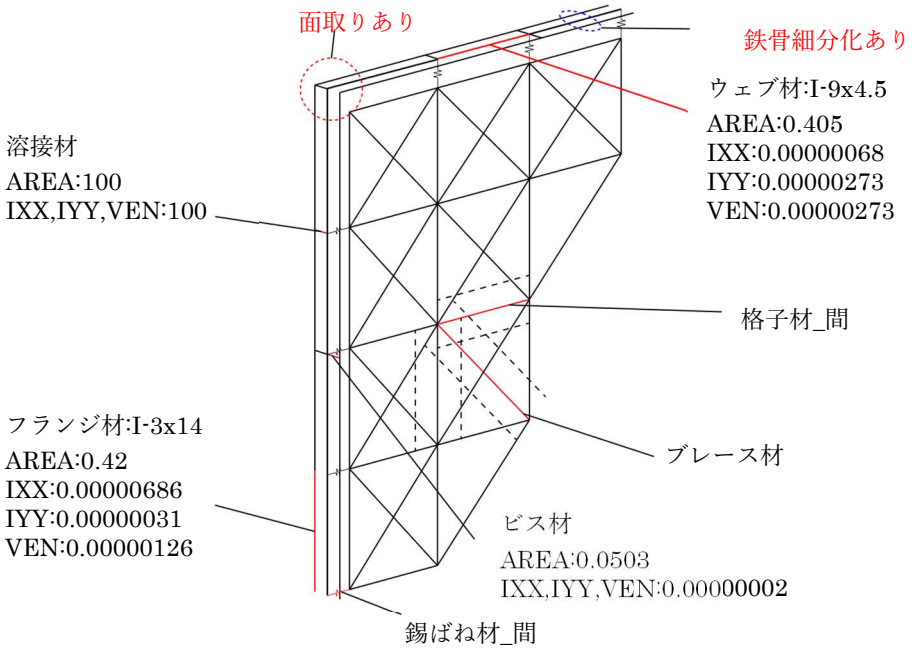


図 4.5 Model103 形状

4.3.3 解析結果

解析の結果についてまとめる。

初期剛性の値は図 4.6 のように Model101 は 13step、Model102 は 10step、Model103 は 11step まで錫ばね除去操作を行い、解析が安定し、初期剛性の値が収束することを確認した。初期剛性の値に関しても Model101 は 1.19（kN/mm）であるが、面取りを施した Model102 では 0.83（kN/mm）となり、さらに鉄骨の細分化を行うことで 0.61（kN/mm）とほぼ実験値と同じ値となった。各解析モデルの初期剛性の値を表 4.2 にまとめる。

小型試験体においては鉄骨の細分化によるモデル化およびガラスの回転拘束位置のモデル化により解析での初期剛性の値が、実験での初期剛性の値に近づくことが確認された。

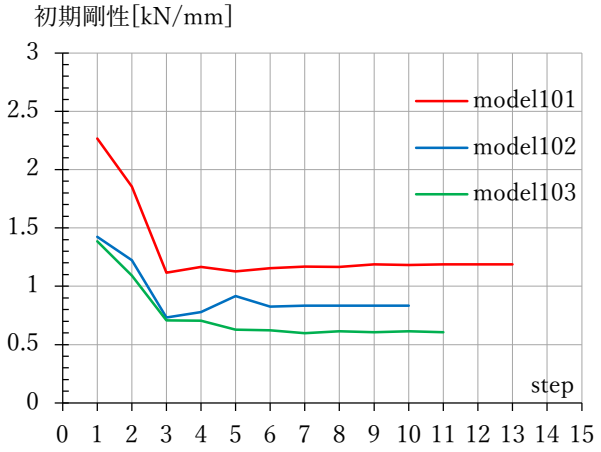


図 4.6 小型試験体の各解析モデルの初期剛性と step 数の関係

表 4.2 小型試験体解析結果まとめ

	メッシュ幅 (mm)	錫の剛性変化	面取り	鉄骨細分化	初期剛性 (kN/mm)
Model101	25	あり	なし	なし	1.16
Model102	25	あり	あり	なし	0.83
Model103	25	あり	あり	あり	0.61
小型試験体 4	-	-	-	-	0.58
小型試験体 5	-	-	-	-	0.53

4.4 卍型試験体の解析

4.4.1 解析モデル種類

卍型試験体の解析モデルも小型試験体と同じくモデル化の手法を組み合わせて複数作成し、面取り及び鉄骨細分化の効果を確認することとした。以下にモデルの種類とモデル化の手法を示す。

- Model1001…試験体をメッシュ幅 25mm で細分化した詳細解析モデルであり、面取り及び鉄骨細分化は施さない。

- Model1002…Model1001 に面取りを施した詳細解析モデル。

- Model1003…Model1001 に面取り及び鉄骨細分化を施した詳細解析モデル。

表 4.3 卍型試験体解析モデルの種類およびモデル化条件

	メッシュ幅 (mm)	錫の剛性変化	面取り	鉄骨細分化
Model1001	25	あり	なし	なし
Model1002	25	あり	あり	なし
Model1003	25	あり	あり	あり

4.4.2 解析モデル形状

Model1001 の形状及び赤丸部分の拡大図を図 4.7 に示す。Model1002,1003 においても同じ位置の拡大図である。メッシュ幅 25mm で試験体を骨組材、ガラス材、錫ばね材に細分化した詳細解析モデルである。面取り及び鉄骨細分化は行わないため、錫ばね材はガラスの角部分にも設けられており、鉄骨は 1 本の骨組材によりモデル化している。小型試験体と同じくガラスの角部分の錫ばね材_端には錫ばね材_間の半分の剛性を与え、ガラスの端の格子材_端は格子材_間の半分の剛性を与えた。実験において初期剛性は荷重が 1kN を超えると安定した値を取るようになるため、解析では解析モデルの上部 4 点に水平荷重を 0.05tf ずつ、合計 1.96kN を与えた時の頂部 2 点の変位の平均から解析の初期剛性を算出した。試験体上部の 4 点の節点条件は x 方向の変位、z 方向の変位、y 軸回りの回転はフリーとし、y 方向の変位、x 軸回りの回転、z 軸回りの回転は固定とした。また、試験体下部の節点条件は両端の 2 点は変位、回転共に全て固定とし、間の 2 点は y 軸回りの回転のみフリーとし、残りは固定とした。Model1002、Model1003 においても与える荷重及び節点条件は同じである。

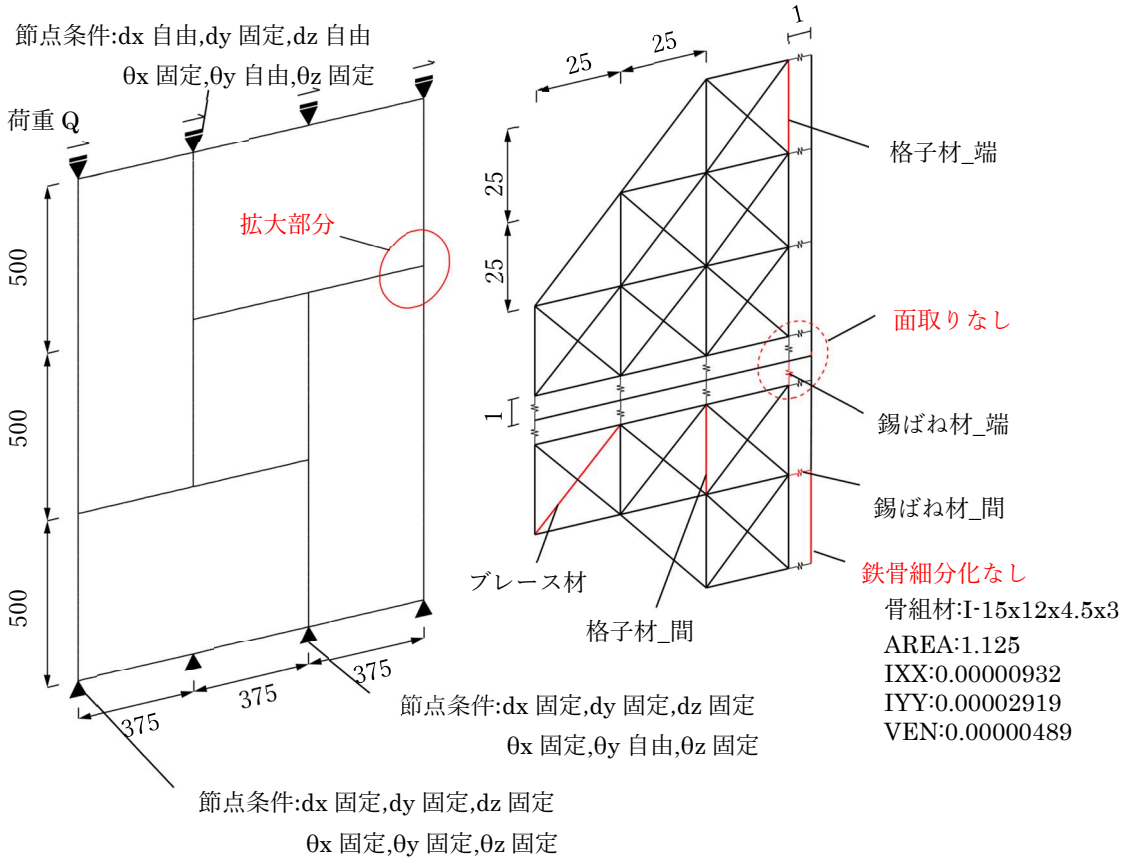


図 4.7 Model1001 形状

Model1002 の形状はに示す形状であり、Model1001 から 5 枚のガラスの角部分の錫ばね材を取り除き面取りを施した解析モデルである。

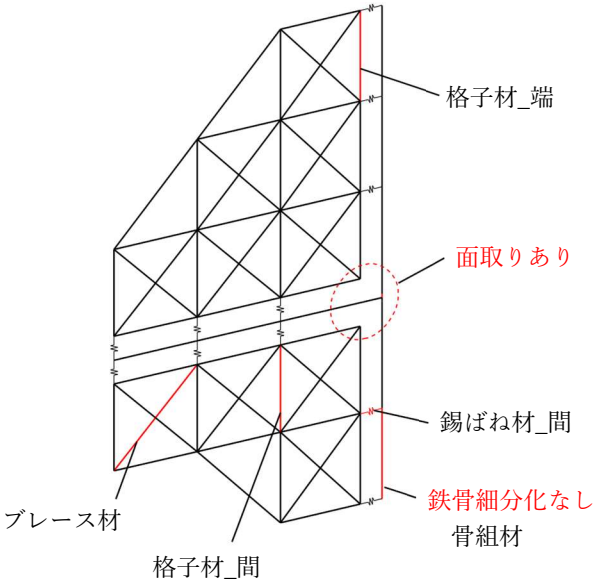


図 4.8 Model1002 形状

Model1003 の形状はに示す形状であり、面取り及び鉄骨の細分化を施している。また、に示すウェブ同士が分離していた位置においてはフランジ材のみを設け、ウェブ材を設けない。

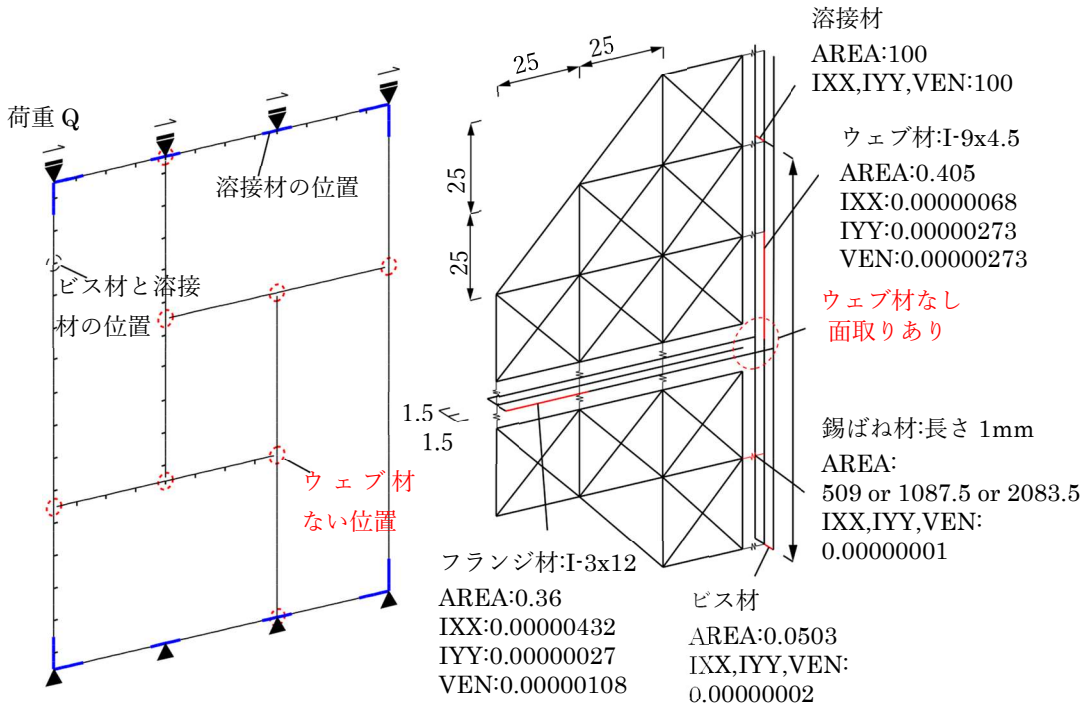


図 4.9 Model1003 形状

4.4.3 解析結果

解析の結果についてまとめる。

初期剛性の値は図 4.10 のように Model1001 は 13step、Model1002 は 11step、Model1003 は 15step まで錫ばね除去操作を行い、解析が安定し、初期剛性の値が収束することを確認した。表 4.4 に各解析モデルでの初期剛性の値と実験値についてまとめる。

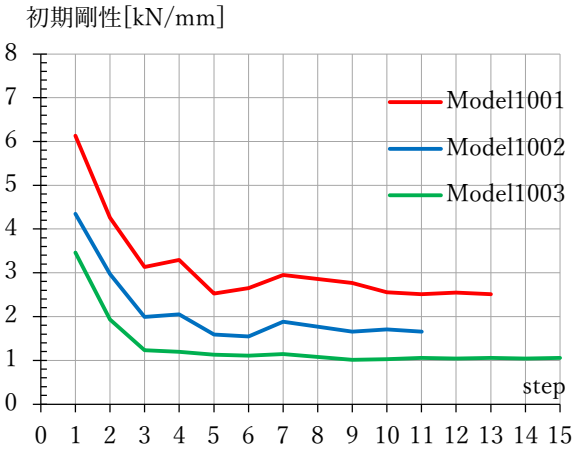


図 4.10 卍型試験体の各解析モデルの初期剛性と step 数の関係

表 4.4 卍型試験体解析結果まとめ

	メッシュ幅 (mm)	錫の剛性変化	面取り	鉄骨細分化	初期剛性 (kN/mm)
Model1001	25	あり	なし	なし	2.51
Model1002	25	あり	あり	なし	1.66
Model1003	25	あり	あり	あり	1.06
卍型試験体 13	-	-	-	-	1.00

既往研究で行ったフランジ幅 12mm の卍型試験体の実験結果から、ガラス 3 およびガラス 4 の回転は図 4.11、図 4.12 に示すようにガラス 3 の方がガラス 4 に比べてより回転して骨組に拘束されていることが分かる。Model1003 の解析後のガラスの位置は図 4.13 に示す位置であり、ガラス 3 の方がガラス 4 より回転して拘束されている様子が再現されている。図 4.13 は解析の変位を 100 倍して表示している。

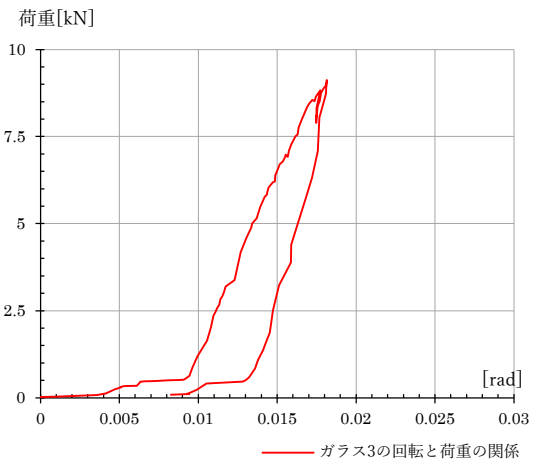


図 4.11 ガラス 3 の回転と荷重の関係

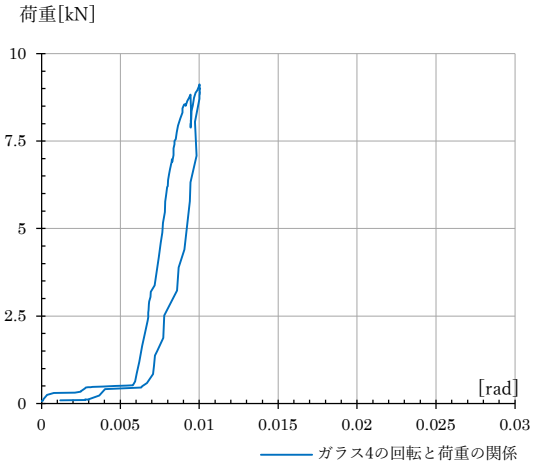


図 4.12 ガラス 4 の回転と荷重の関係

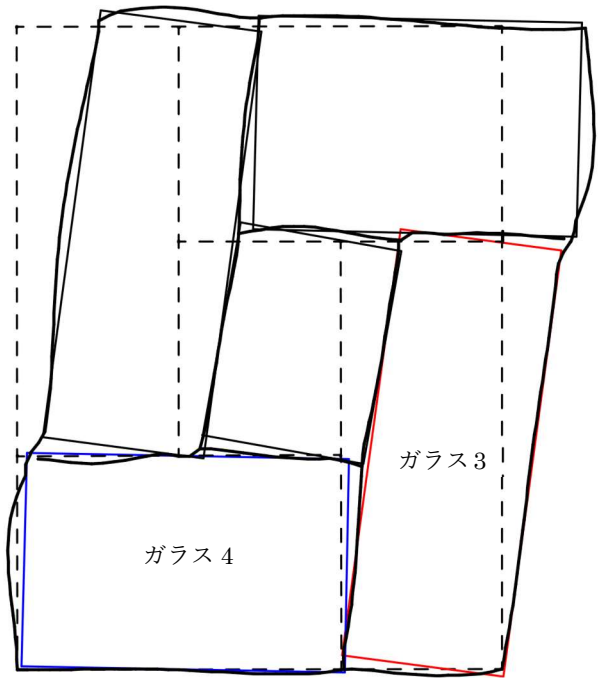


図 4.13 解析による変形

錫ばね材は図 4.14 に示す位置において剛性を発揮しており、ガラス 1 の左上の角の変形後の各部材の位置関係を図 4.15 に示す。部材番号 2609,2610 の錫ばね材は骨組材側の節点位置とガラス材側の節点位置にずれが生じており、錫ばねにせん断力が発生している可能性がある。表 4.5 に部材番号 2609 と 2610 の応力情報をまとめる。曲げモーメントは発生していないが、面内方向のせん断力は軸力の 0.1%に満たない大きさではあるが発生している。本解析では初期剛性の値を得るための解析であり、1.96kN の荷重に対して変形が 1.85mm と小さいためせん断力は大きくなかったが、今後終局までの解析を行う場合には、変形が 30mm を超えるため、錫ばね材に働くせん断力の大きさが大きくなり、解析に影響を与える可能性がある。

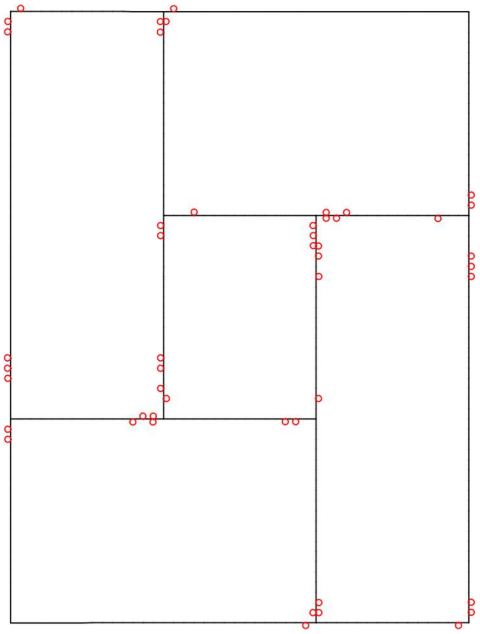


図 4.14 錫ばね位置(Model1003)

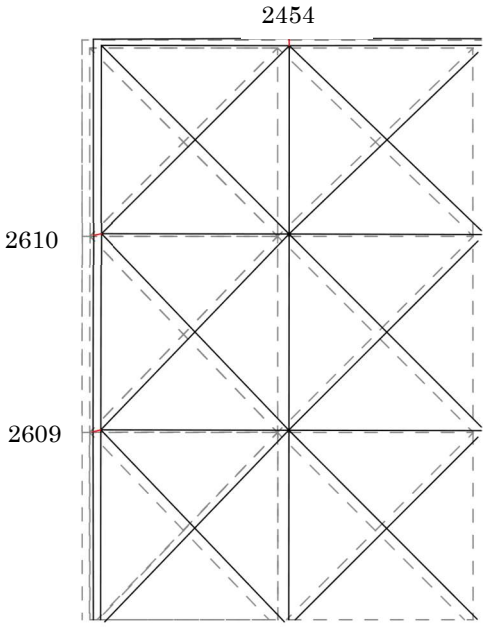


図 4.15 ガラス 1 左上角部分の錫ばねの変形

表 4.5 錫ばね材に働く応力

部材番号	軸力(tf)	面外方向 せん断力(tf)	面内方向 せん断力(tf)	曲げ モーメント
2609	0.02627	0.00000	-0.00003	0.00000
2610	0.08576	0.00000	-0.00003	0.00000

4.5 まとめ

小型試験体及び卍型試験体において鉄骨の細分化によるモデル化、面取りによるガラスの拘束位置のモデル化の有効性について確認し、詳細解析モデルの解析により実験の初期剛性を再現できることを示した。

一方で、骨組材とガラス材の間を 1 本のみでモデル化した詳細解析モデルでは終局までの解析を行う場合、変形が大きくなるに伴い錫ばね材に働くせん断力が大きくなってしまう可能性がある。

次章では、今井らにより提案されたガラスと錫の隙間のモデル化による解析モデルを作成し解析を行う。

第5章 錫ばね剛性判定法による解析

5.1 概説

本章では 5.2 節で述べる錫ばね剛性判定法による解析を行い、得られた知見と課題についてまとめる。

5.2 節では、本章では既往研究 [12]において提案されたガラスの位置変化を再現するモデル化の手法及び錫ばね剛性判定法の解析方法について述べる。

5.3 節では、卍型試験体の解析モデルにおいて行った錫ばね剛性判定法による解析について述べる。

5.4 節では、解析からガラスが拘束されていく様子についてまとめる。

5.5 節では、解析において得られた課題についてまとめる。

5.2 錫ばね剛性判定法

5.2.1 ガラスの位置変化のモデル化

4.5 で述べたように骨組材とガラス材の間を 1 本の錫ばねでモデル化した解析モデルでは、変形と共に錫ばね材に働くせん断力が大きくなってしまう可能性がある。したがって、既往研究[12]では図 5.1 のようにガラス側の端点から骨組材に向けて複数の錫ばね材を設置したモデル化が提案されている。

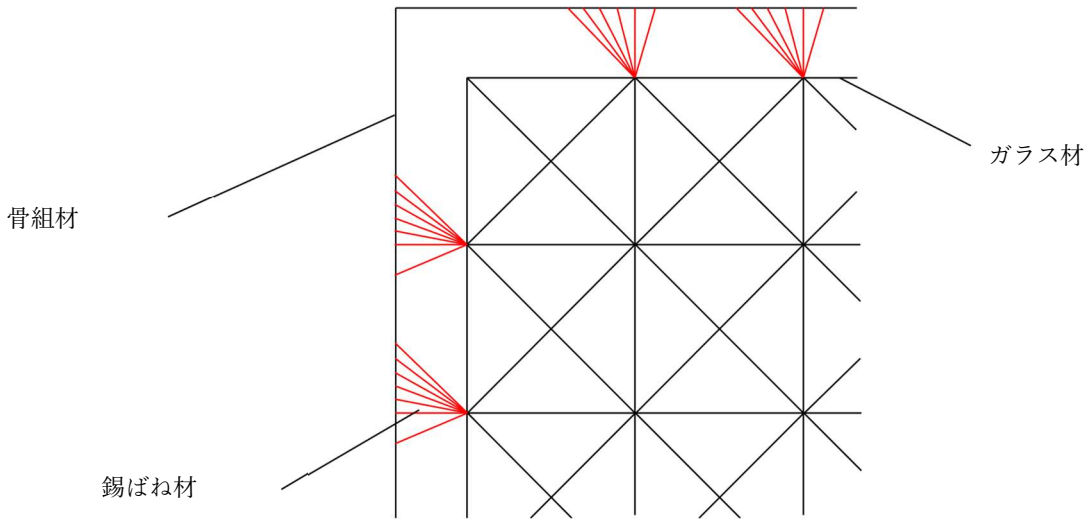


図 5.1 錫ばね材を複数設けた解析モデル

5.2.2 錫ばねの剛性選択

図 5.1 に示したモデル化による解析モデルでは、最初の解析での錫ばね材の剛性は限りなく小さくし、微小荷重を与えることで解析モデルを変形させる。変形した後のガラス材と骨組材の位置関係に応じて錫ばね材に与える剛性を以下に示すフローチャートのように決定する。

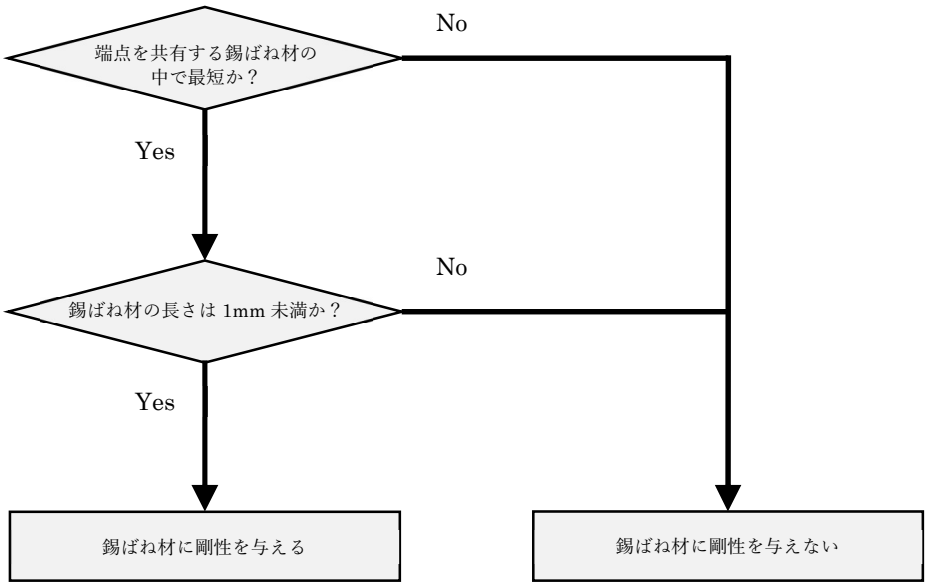


図 5.2 錫ばね材の剛性判定フローチャート

図 5.2 に示した剛性判定を解析ステップ終了後に毎回行い、新しい解析を行うことで図 5.3 のように剛性が与えられ、有効な錫ばね材として働く錫ばね材はガラスと骨組材の位置関係に応じて変していく。

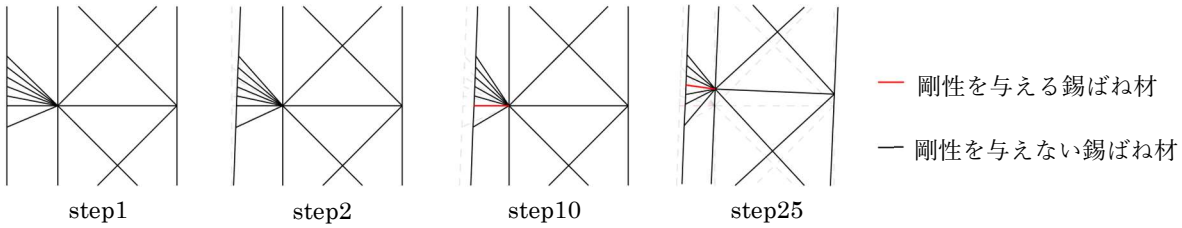


図 5.3 有効な錫ばね材がガラスと骨組材の位置関係に応じて変化してく様子

5.2.3 錫の材料非線形性との組み合わせ

本研究では、既往研究で行われていた錫ばねの剛性選択に錫の材料非線形性を組み込み、錫ばね材の縮みが大きくなるにつれてより大きい剛性を与えた。剛性判定のフローチャートを図 5.4 に示す。縮みの大きさによって与える錫ばね材のばね定数 k_1,k_2,k_3 を表 5.1 にまとめる。

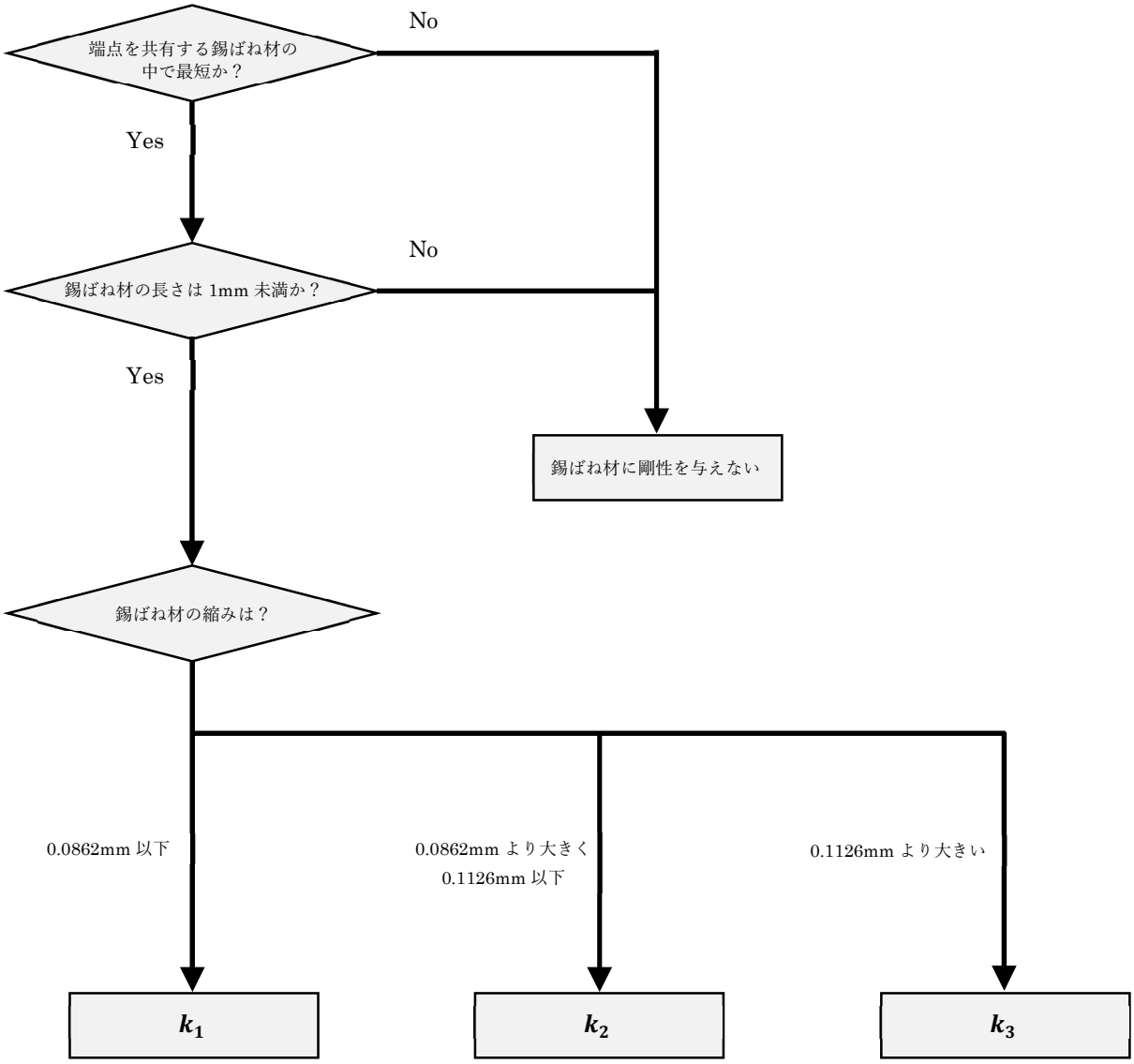


図 5.4 錫の材料非線形性を組み合わせた錫ばねの剛性判定フローチャート

表 5.1 錫ばねの剛性対応表

錫ばねの縮み (mm)	0＜Δtin ≦0.0862	0.0862＜Δtin ≦0.1126	0.1126＜Δtin
錫のばね定数 (kN/㎜)			

5.3 卍型試験体の解析

卍型試験体のガラスの位置変化をモデル化した詳細解析モデルにおいて、錫ばね剛性判定法による解析を行う。詳細解析モデルのメッシュ幅は 25mm とし、面取りを施している。正方向載荷用の解析モデルと負方向載荷用の解析モデルを作成し解析を行った。

5.3.1 解析モデル形状

正方向載荷の解析モデル(Model301)は図 5.5 に示す形状であり、荷重は上部 4 点に均等に加える。負方向載荷の解析モデル(Model302)では図 5.5 において上部 4 点に図とは逆の方向に荷重を与える。変形は解析モデル上部の両端の点の変位から求める。正方向の解析モデルの四隅の拡大図を図 5.7 に示す。下側のガラスと鉄骨の隙間を 1mm としているため、上側のガラスと鉄骨の隙間は 3.5mm となっている。また、左右のガラスと鉄骨の隙間は 2.25mm である。正方向載荷では左上の角と右下の角でガラスが拘束され、右上と左下の錫ばね材は剛性を発揮しないことが予想されるため、右上と左下の角の錫ばね材の本数は減らしている。また、同様に負方向載荷では左上と右下の錫ばね材は剛性を発揮しないことが予想されるため、錫ばね材の本数を減らしている。

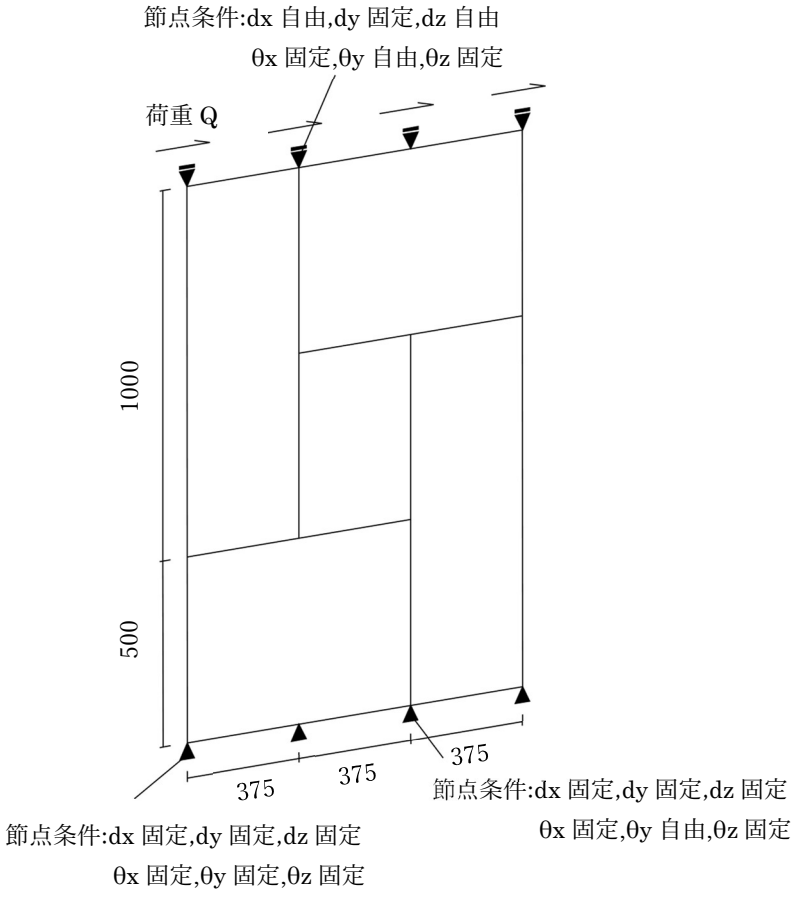


図 5.5 詳細解析モデル概形(Model301)

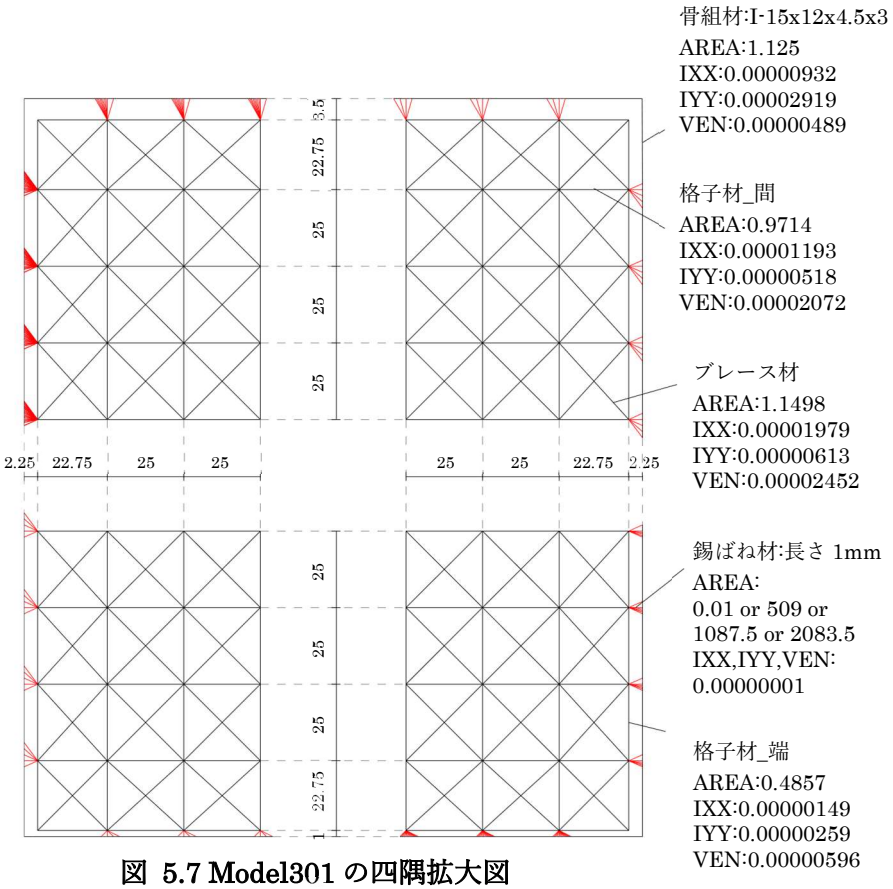


図 5.7 Model301 の四隅拡大図

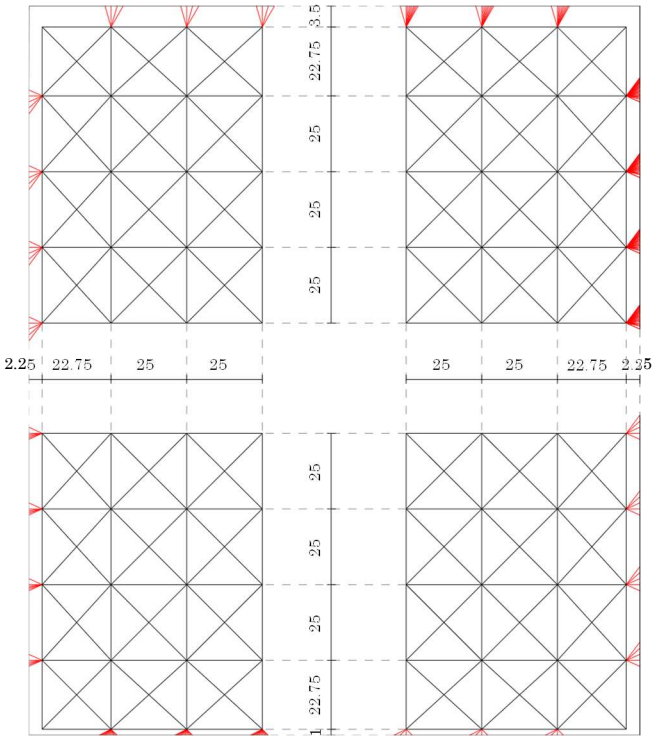


図 5.6 Model302 の四隅拡大図

5.3.2 解析経過

正方向載荷の解析で各ステップに解析モデルの上部 4 点に与えた荷重の合計を表 5.2 に負方向載荷の解析で各ステップに与えた荷重の合計を表 5.3 に示す。途中で与える荷重を小さくしていることについては後述する。

表 5.2 各 step において加える荷重(正方向)

step 数	荷重(tf)
1	0.0002
2	0.00008
3~4	0.0002
5~7	0.00004
8~10	0.0006
11~15	0.001
16~20	0.0006
21~25	0.001
26~30	0.0012
31~34	0.002
35~36	0.004
37~40	0.006
41~44	0.01
45~50	0.02
51~55	0.04

表 5.3 各 step において与えた荷重(負方向)

step 数	荷重(tf)
1~10	-0.00004
11~15	-0.00008
16~24	-0.0002
25~29	-0.00004
30~38	-0.0002
39~43	-0.0004
44~48	-0.0006
49~51	-0.001
52~53	-0.002
54~58	-0.003
59	-0.004
60~63	-0.001
64~68	-0.003
69~71	-0.004
72~78	-0.008
79~80	-0.02
81~85	-0.04

正方向載荷の解析で錫ばね材が効いて剛性を発揮していく様子を示す。錫ばね材の縮みの長さに応じて図 5.8 に示す 3 色により表示する。

● … 0＜Δtin ≦0.0862

● … 0.0862＜Δtin ≦0.1126

● … 0.1126＜Δtin

図 5.8 縮みに応じて表示する色

Step2
荷重…0.00196kN,変形…0.28mm
各ガラスの右下の錫ばねが効く

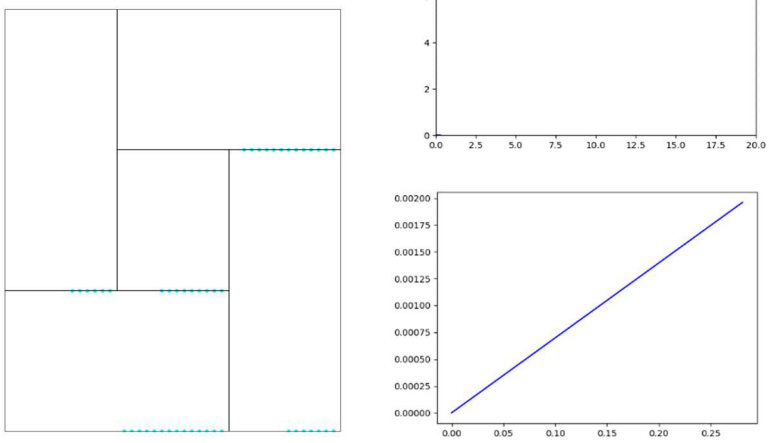


図 5.9 step2 での錫ばね位置と荷重変形曲線

Step5
荷重…0.00706kN,変形…1.71mm
ガラス 1 の左上の横側に錫ばね

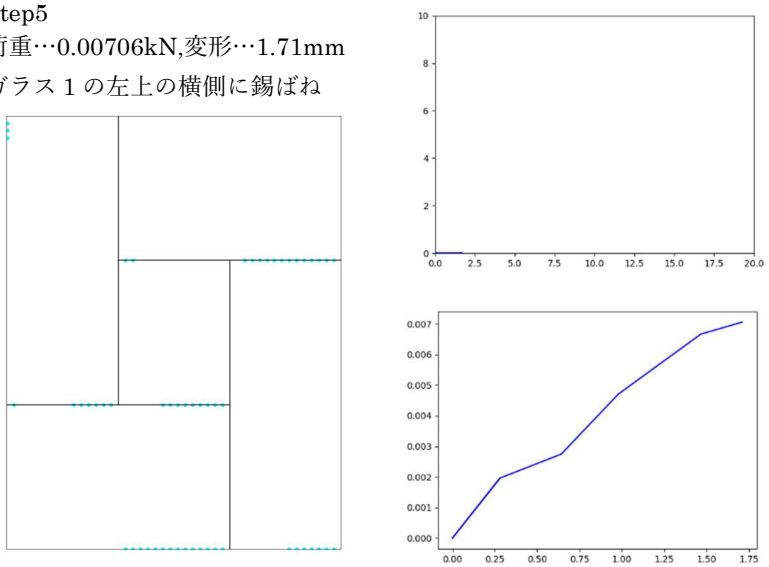


図 5.10 step5 での錫ばね位置と荷重変形曲線

Step10
荷重…0.0255kN,変形…3.57mm
ガラス 2,3 の左上の横側に錫ばね

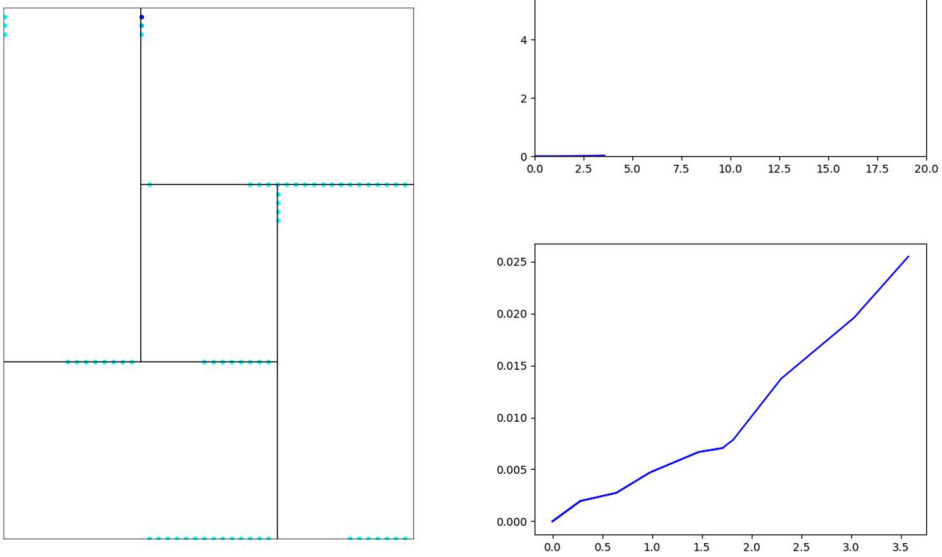


図 5.11 step10 での錫ばね位置と荷重変形曲線

Step12
荷重…0.0451kN,変形…5.25mm
全てのガラスの左上横側に錫ばね

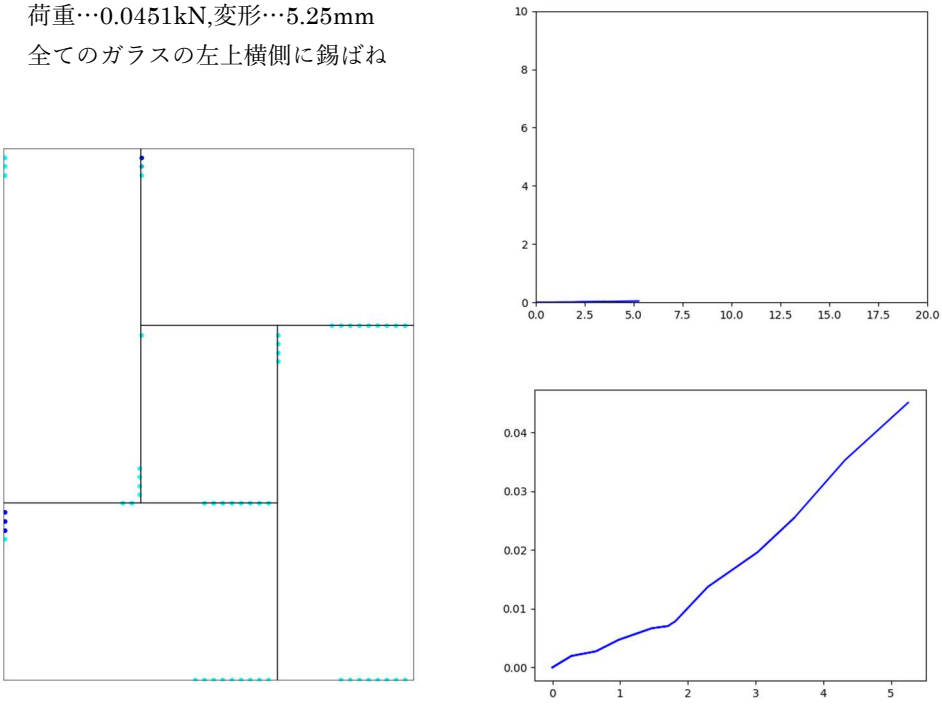


図 5.12 step12 での錫ばね位置と荷重変形曲線

Step24
荷重…0.143kN,変形…11.59mm
全てのガラスの右下が拘束される

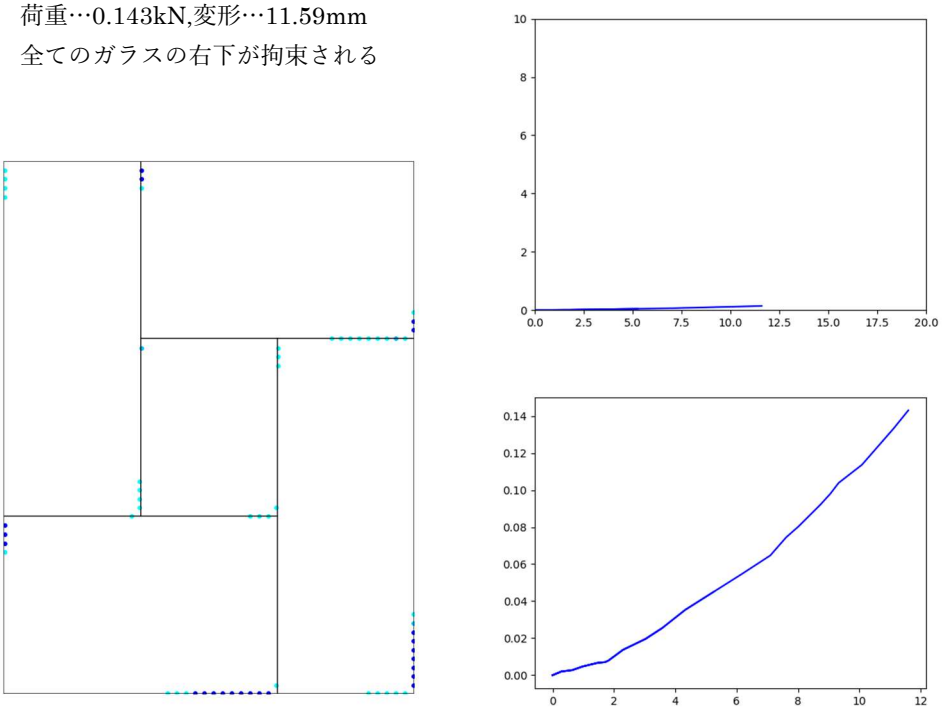


図 5.14 step24 での錫ばね位置と荷重変形曲線

Step26
荷重…0.165kN,変形…12.23mm
ガラス 1 の左上の上側の錫ばねが剛性を発揮し、ガラス 1 が剛性を発揮する。

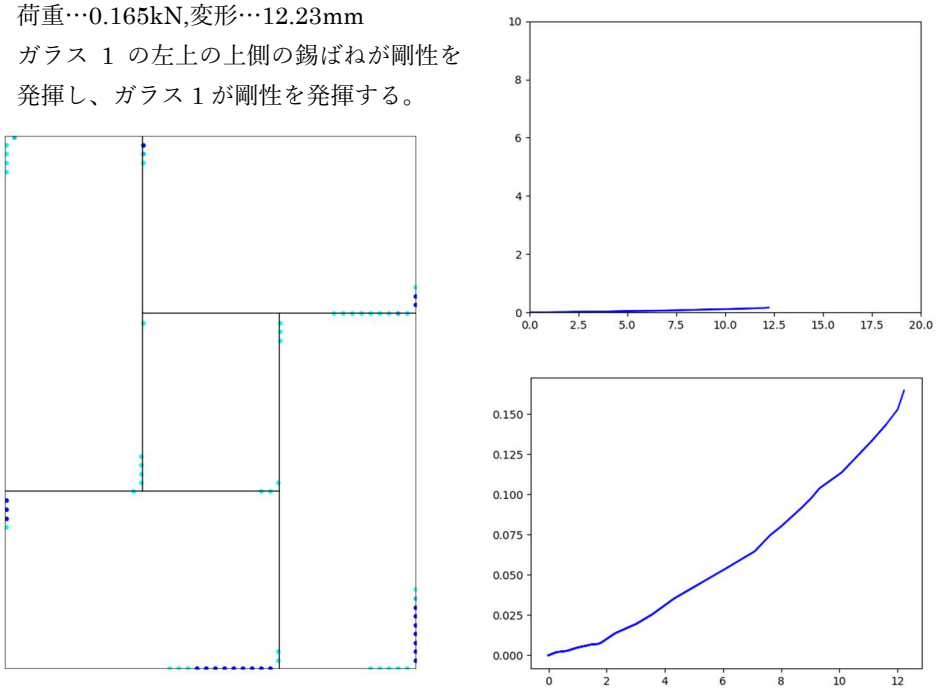


図 5.13 step26 での錫ばね位置と荷重変形曲線

Step31
荷重…0.231kN,変形…13.91mm
ガラス 5 の左上の上側の錫ばねが剛性を発揮し、ガラス 5 が剛性を発揮する。

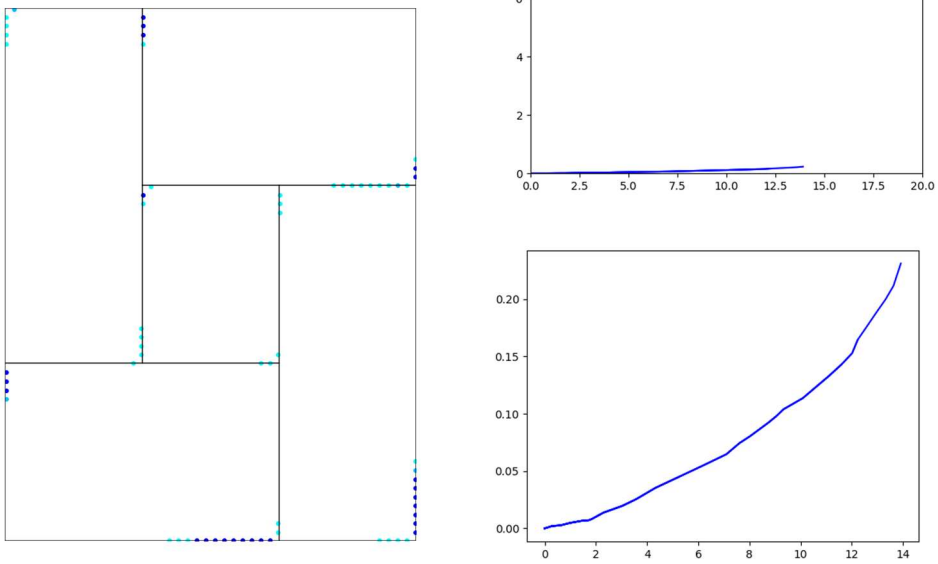


図 5.15 step31 での錫ばね位置と荷重変形曲線

Step34
荷重…0.290kN,変形…14.39mm
ガラス 2 の左上の上側の錫ばねが剛性を発揮し、ガラス 2 が剛性を発揮する。

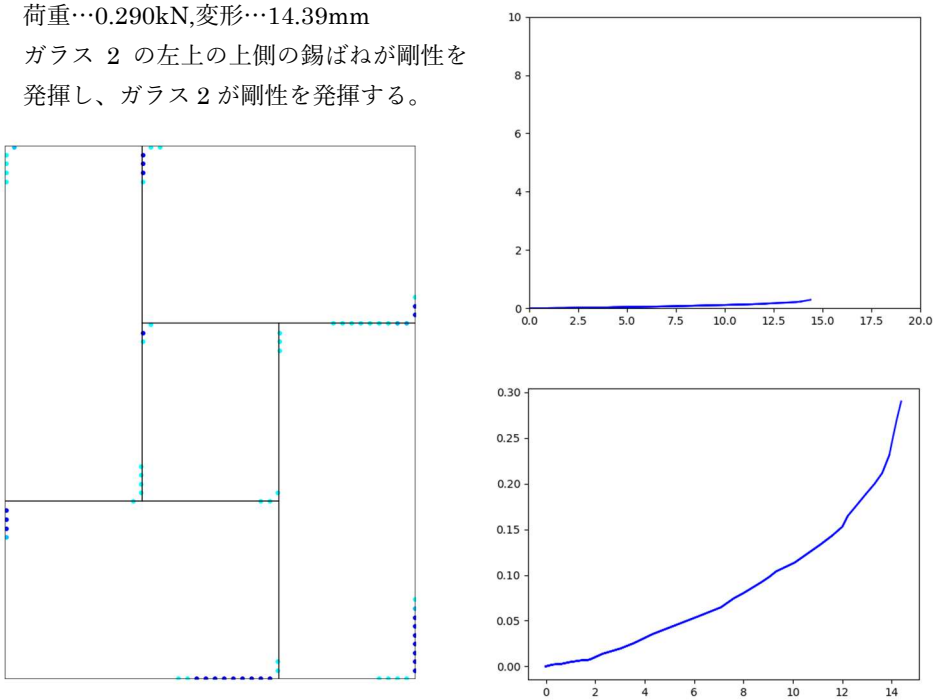


図 5.16 step34 での錫ばね位置と荷重変形曲線

Step38
荷重…0.486kN,変形…15.49mm
ガラス 3 の左上の上側の錫ばねが剛性を発揮し、ガラス 3 も剛性を発揮する。

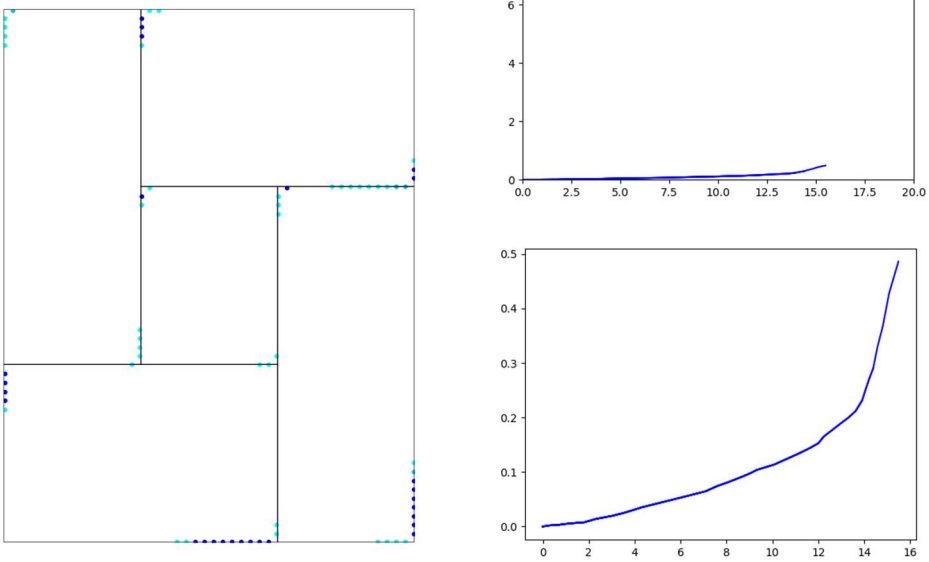


図 5.17 step38 での錫ばね位置と荷重変形曲線

Step41
荷重…0.702kN,変形…15.86mm
ガラス 4 の左上の上側の錫ばねが剛性を発揮し、全てのガラスが剛性を発揮する。

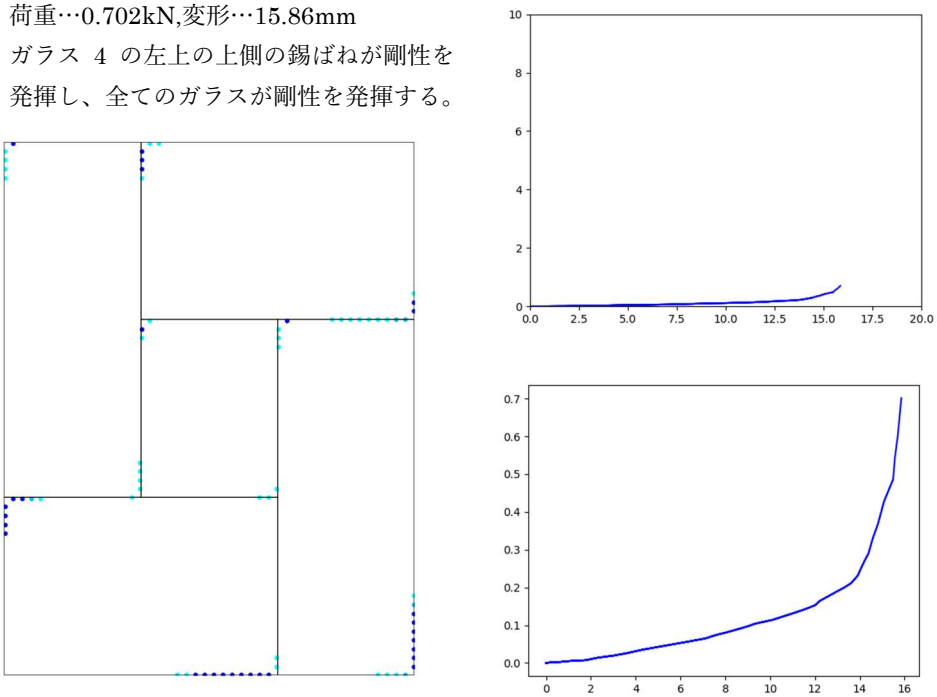
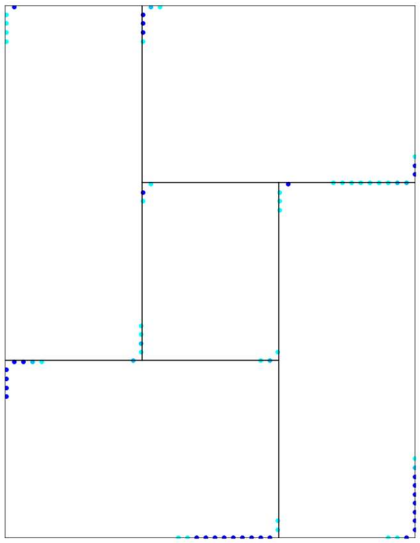
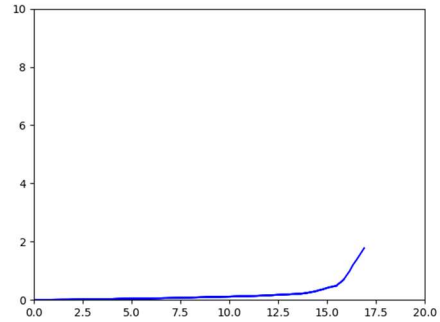


図 5.18 step41 での錫ばね位置と荷重変形曲線

Step48
荷重…1.78kN,変形…16.90mm
グラフが十分立ち上がっている様子が分かる。





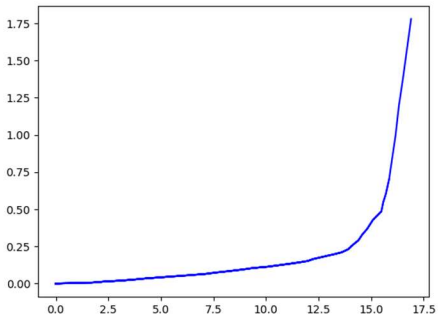
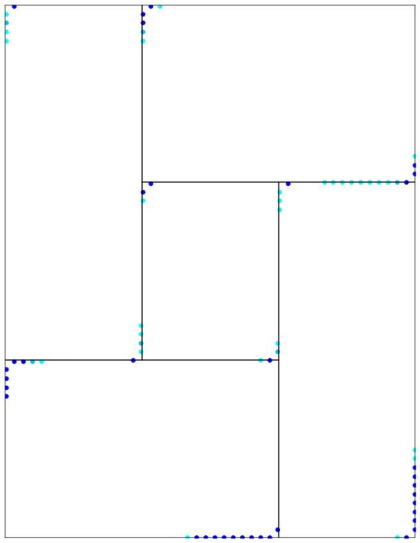
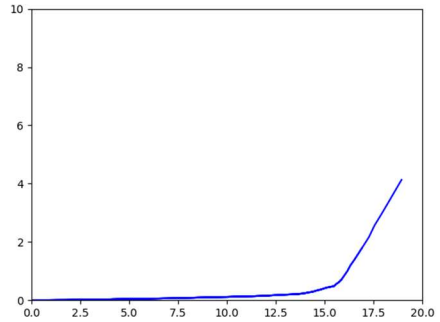


図 5.19 step48 での錫ばね位置と荷重変形曲線

Step55
荷重…4.13kN,変形…18.94mm
グラフが十分立ち上がり、解析を終了。





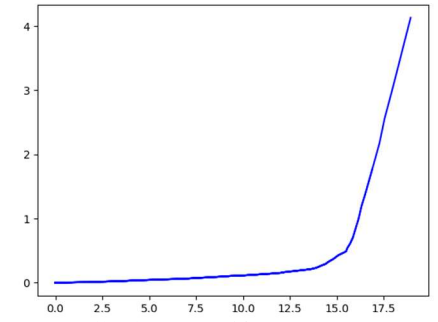


図 5.20 step55 での錫ばね位置と荷重変形曲線

図 5.9～図 5.20 に示した剛性を発揮している錫ばねの分布と荷重変形曲線における剛性が立ち上がる様子から、ガラスの左上の上側の錫ばねが効き始めると、そのガラスが拘束され、剛性を発揮し始めることが分かる。今回の解析では、ガラス 1、ガラス 5、ガラス 2、ガラス 3、ガラス 4 の順に拘束される。それぞれのガラスが拘束されるステップでの剛性値の変化を表 5.4 にまとめる。各ステップでの剛性値は式(5.1)により求めた。表 5.4 からガラスが拘束されていくにしたがって剛性が発揮されていくことが分かる。また、全てのガラスが拘束されることで剛性が急激に上昇することもわかる。

表 5.4 剛性値の変化

step	剛性値 (kN/mm)	解析モデル上の変化
25	0.013	拘束されたガラスなし
26	0.019	ガラス 1 が拘束される
31	0.068	ガラス 5 が拘束される
34	0.111	ガラス 2 が拘束される
38	0.144	ガラス 3 が拘束される
41	0.631	ガラス 4 が拘束される
48	1.037	剛性が安定する

5.3.3 解析結果

解析により得られた荷重変形曲線を図 5.21 に示す。剛性が発揮され始めるまでのスリップ領域は解析の方が大きくなっている。しかし、解析で剛性の発揮され始める位置を実験と合わせるようにグラフを移動させた図 5.22 を見ると、初期剛性に関しては実験と良い一致を見せている。

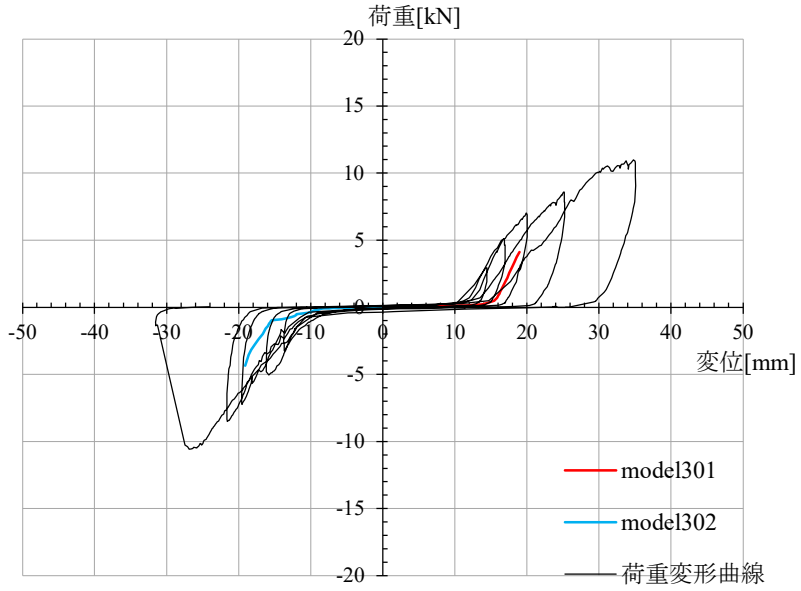


図 5.21 解析による得られる荷重変形曲線と実験での荷重変形曲線

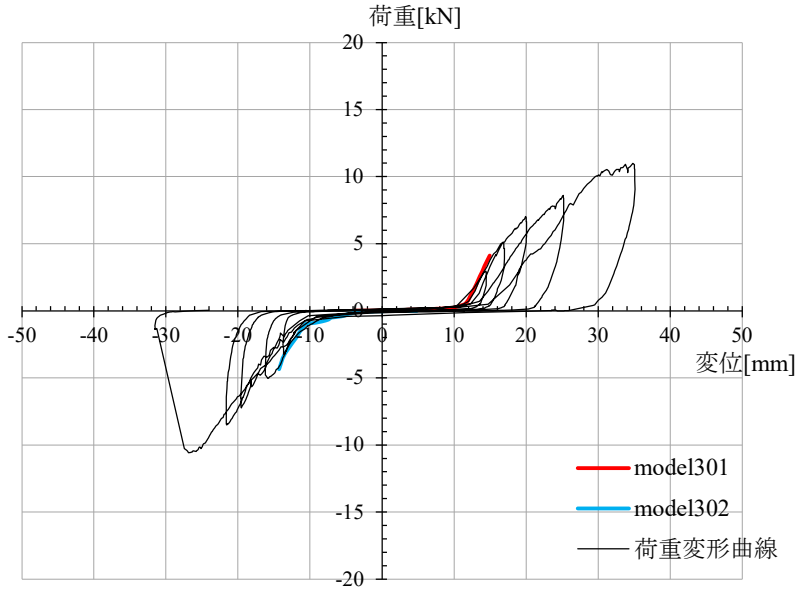


図 5.22 解析による荷重変形曲線を実験の荷重変形曲線に合わせたグラフ

5.4 卍型試験体におけるガラスの位置変化

本解析において、ガラスは骨組に拘束されるまでは錫ばね材からの微小な力を受けて骨組内を平行に移動したり、回転したりとその位置を変化させる。錫ばね材が効いている状態に応じて解析のフェーズを以下の図 5.23 に示すように分ける。

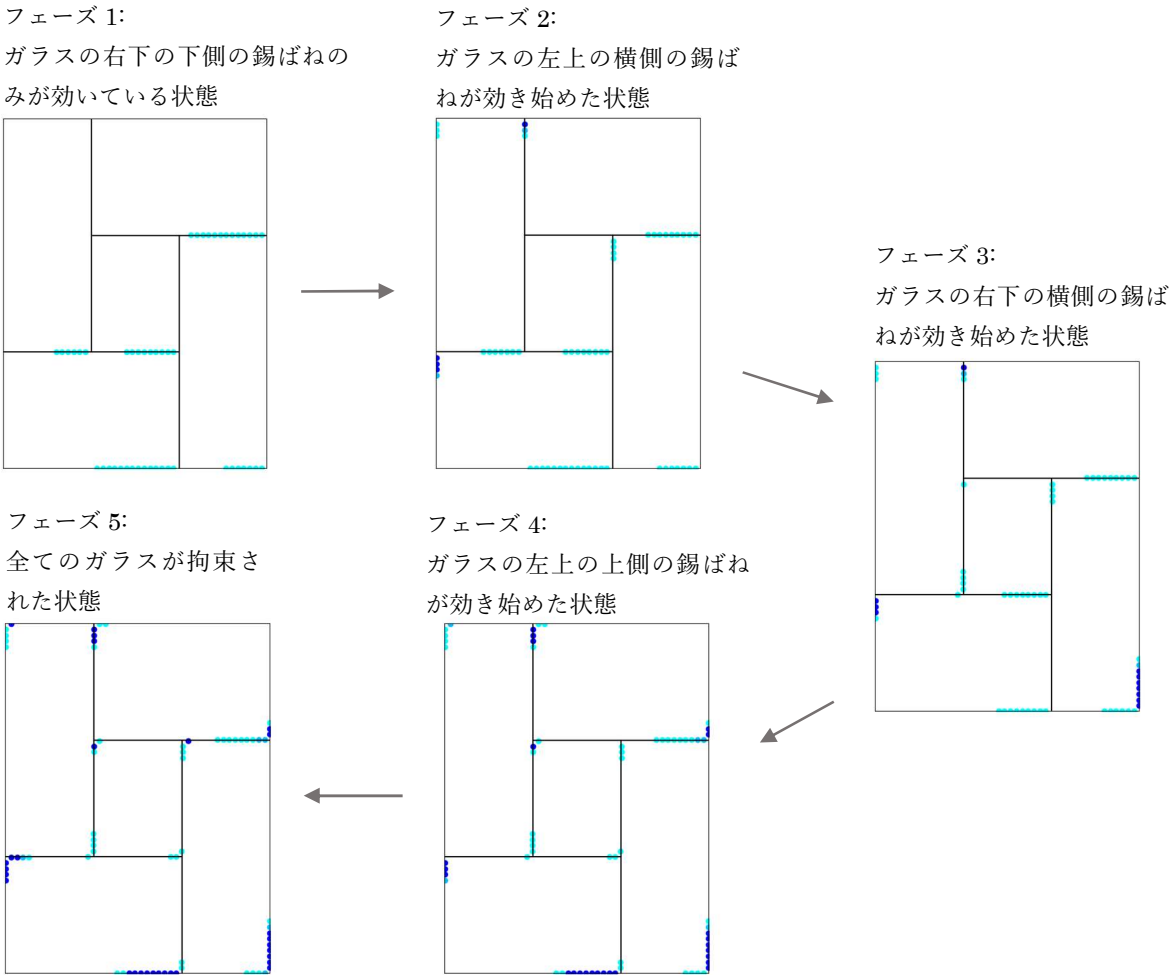


図 5.23 解析のフェーズ

step1 の解析モデルで下側のガラスと骨組の間を 1mm にしているため、step2 の解析からはガラスの右下の角の下側の錫ばね材に剛性が入る。その後フェーズ 1 では、骨組の変形が進み、錫ばね材に正方向載荷では左上の角の横側のガラスと骨組の隙間が 1mm 未満となって錫ばね材に剛性が入る。左上の横側の錫ばね材が効き始めるとフェーズ 2 に移行し、ガラスは水平方向に移動し、次は右下の横側の錫ばね材に剛性が入る。フェーズ 3 ではガラスは回転移動を始め、左上の角の上側の錫ばね材が最後に効きフェーズ 4 となる。そしてガラスが完全に拘束されたフェーズ 5 では解析モデルの剛性が発揮される。このガラスが移動して拘束されていく流れは、既往研究 [12)]において小型試験体で観察された流れと同じであり、他の直線で区切られた模様においても同様にガラスが拘束されていくと考えられる。

5.5 解析における課題

解析を行った際に得られた解析手法の課題としては 2 点を挙げられる。1 つ目は、ガラスは拘束されて剛性を発揮し始める前の解析では骨組内をゆっくりと移動するのであるが、その際に剛性を持って効いていた錫ばね材の剛性が失われてしまう時があること。2 つ目は、錫ばね材が剛性を発揮する直前で荷重を加えすぎてしまうと錫ばね材がかなり縮んだ状態で効き始めてしまうことである。表 5.2、表 5.3 において荷重が前のステップと比較して不自然に小さくなっているステップがあるのはこの 2 つの問題があったためである。

1 つ目の課題は解析の後の変形した位置が図 5.24 のようにガラスと鉄骨の間の隙間は 1mm 未満で本来錫ばね材に剛性を与えなければならないが、ちょうど鉄骨側の端点から 1mm 以上の所に位置しているために剛性が与えられなかったということが原因で生じている。本研究では上記のように錫ばね材の剛性が失われてしまった場合は、「最短の錫ばね材に剛性を入力する」あるいは「直後の解析からは与える荷重を小さくして錫ばね材が過度に圧縮されることなく再び剛性を持つまで変形させる」といった 2 つの方法で対応した。

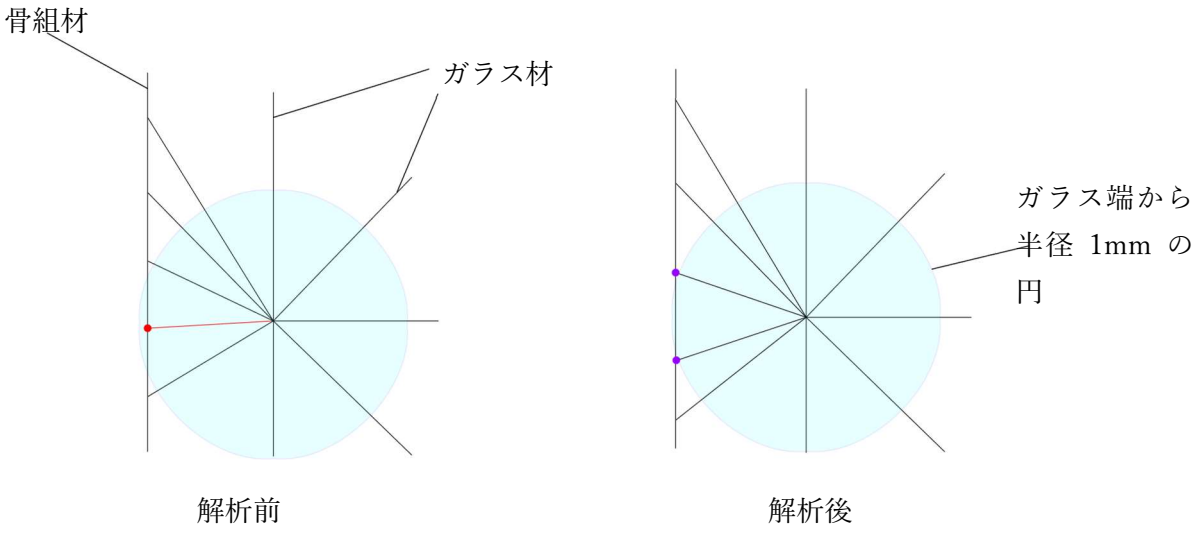


図 5.24 錫ばね材に剛性が入力されない位置

また、どこか 1 つの錫ばね材でのみ剛性入力されなかった場合は他の錫ばね材のおかげで錫ばね材が急激に縮んでしまったり、ガラスが骨組を突き抜けてしまったりすることは防がれるが、のガラス 2 の左下のように剛性が入力されないという状況はある 1 辺の錫ばね材全てに対して起きることが多かった。

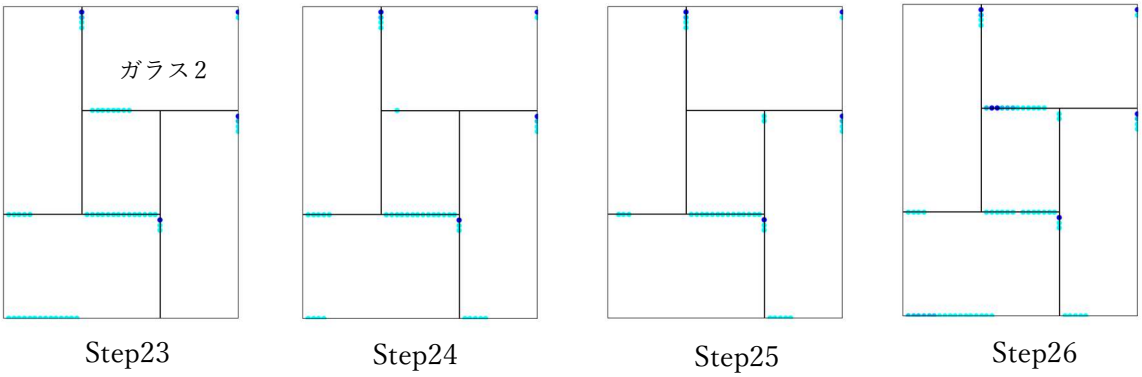


図 5.25 剛性が入力されない錫ばね材の発生例(負方向解析)

また、解決策としては、錫ばね材のピッチをさらに細かくすることで図 5.24 の錫ばね材に剛性が入力されない位置にガラス側の端点が来てしまう確率を下げる方法や、鉄骨側の錫ばね材の端点のピッチをランダムにすることで錫ばね材が全て同時に図 5.24 の位置に来てしまうことを防ぐ方法が考えられる。

2 つ目は、1 ステップ前の解析でガラスと骨組の隙間が限りなく 1mm に近くなっているにも関わらず、荷重を加えすぎると錫ばね材が急激に押しつぶされてしまうことである。本研究では荷重を小さくして解析をやり直すことで対応した。この問題に関しては、現段階では解析は毎回錫ばねの入り具合を確認しながら行っているため、解析回数を減らすために入力する荷重を増加させているが、解析を自動化し荷重を微小な値で固定することでこの問題は解決される。

第6章 結論

6.1 まとめ

本研究では、初期剛性を再現するモデル化の手法として、ガラスの拘束位置のモデル化及び鉄骨の細分化を用いてモデル化したメッシュ幅 25mm の詳細解析モデルにおいて錫ばね除去法による解析を行うことで、小型試験体、卍型試験体ともに実験での初期剛性を再現できることを示した。

錫ばね剛性判定法による解析では、ガラスの拘束位置及びガラスの位置変化のモデル化を用いたメッシュ幅 25mm の詳細解析モデルにおいて、実験での初期剛性と比較的近い大きさの初期剛性を発揮する荷重変形曲線を描くことができることを示し、卍型試験体においても小型試験体と同様にガラスが位置変化して拘束されることを確認した。また、錫ばね剛性判定法による解析での課題についても触れ、解決方法についての提案を行った。

6.2 今後の課題

まず本研究では、鉄骨細分化を行った詳細解析モデルにおいて錫ばね剛性判定法による解析を、途中でガラスが骨組から飛び出すことなく行った場合に描かれる荷重変形曲線に関しては未確認である。

今後の課題としては、小型試験体や卍型試験体のような直線で区切られた模様ではない曲線で区切られた模様のステンドグラス構造体において、本研究で提案したモデル化の手法を用いてモデル化した解析モデルで初期剛性を再現可能であるかの確認や、終局までの荷重変形曲線を再現する解析への応用などがあげられる。

付録

1.錫ばね除去プログラム

本論文における線形解析を行う際に 1 つ前の解析モデルのデータと解析結果のデータから次の解析モデルのデータを作成するプログラムである。副プログラムでは初期剛性の計算及び圧縮の入った錫ばねの位置の表示を行った。

主プログラム

```
#フォルダへの移動
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')

import numpy as np
import math

#ファイル名、step 数、錫ばねの長さ指定
file_name='kogata14mm_5125_'
step=6
tin=1

next_step=step+1
file_input_1=file_name+str(step)+'.inp'
file_input_2=file_name+str(step)+'.otp'
file_output=file_name+str(next_step)+'.inp'

#データの読み込み
data_inp=[]
data_otp=[]
for line in open(file_input_1, "r"):
    read_inp=line.split()
    data_inp.append(read_inp)
open(file_input_1, "r").close

for line in open(file_input_2, "r"):
    read_otp=line.split()
    data_otp.append(read_otp)
open(file_input_2, "r").close

x=16+int(data_inp[3][1])*5+1+(int(data_inp[4][1]))*15+1
y=6+int(data_inp[2][1])*2+6
z=x+int(data_inp[1][1])+1

#錫ばねに入力された応力度に応じた錫ばねの剛性判定
for i in range(int(data_inp[2][1])):
    if 400<int(data_inp[z+i][3])<500:
        if (float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[180+(int(data_inp[z+i][3])-402)*15][1])*1000*1000/1000000000)<=10.989:
            data_inp[z+i][3]='402'
        elif 10.989<(float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[180+(int(data_inp[z+i][3])-402)*15][1])*1000*1000/1000000000)<=20.593:
            data_inp[z+i][3]='403'
        elif 20.593<(float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[180+(int(data_inp[z+i][3])-402)*15][1])*1000*1000/1000000000):
            data_inp[z+i][3]='404'
    if 500<int(data_inp[z+i][3])<600:
        if (float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[240+(int(data_inp[z+i][3])-502)*15][1])*1000*1000/1000000000)<=10.989:
            data_inp[z+i][3]='502'
        elif 10.989<(float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[240+(int(data_inp[z+i][3])-502)*15][1])*1000*1000/1000000000)<=20.593:
            data_inp[z+i][3]='503'
        elif 20.593<(float(data_otp[6+2*(int(data_inp[z+i][1])-1001)][3])*9.8*1000)/(float(data_inp[240+(int(data_inp[z+i][3])-502)*15][1])*1000*1000/1000000000):
            data_inp[z+i][3]='504'

#ELEM データ以外のデータの読み込み
other_data=[]
for i in range(z-1):
    other_data.append(data_inp[i])

#引張の入った錫ばね材とその他の材の分別
n_elem=int(data_inp[2][1])
new_elem_num=[]
counter=-1
elem_counter=[]

del_elem=[]

for i in range(n_elem):
    counter=counter+1
    if int(data_otp[6+2*i][1])//100<4:
        new_elem_num.append(data_otp[6+2*i][0])
        elem_counter.append(counter)

    elif int(data_otp[6+2*i][1])//100>=4 and float(data_otp[6+2*i][3])>=0:
        new_elem_num.append(data_otp[6+2*i][0])
        elem_counter.append(counter)
    else:
        del_elem.append(data_otp[6+2*i][0])

#新しい部材データ
new_elem=[]
for i in range(len(elem_counter)):
    new_elem.append(data_inp[z+elem_counter[i]])

#データの結合
other_data[2][1]=str(len(new_elem_num))
other_data_edit_1=[]
for i in range(len(other_data)):
    other_data_edit_1.append(other_data[i])
for i in range(len(other_data_edit_1)):
    other_data_edit_1[i]=" ".join(other_data_edit_1[i])
other_data_edit='¥n'.join(other_data_edit_1)

new_elem_edit_1=[]
for i in range(len(new_elem)):
    new_elem_edit_1.append(new_elem[i])
for i in range(len(new_elem_edit_1)):
    new_elem_edit_1[i]=" ".join(new_elem_edit_1[i])
new_elem_edit='¥n'.join(new_elem_edit_1)
new_inp=other_data_edit+'¥n'+'¥n'+new_elem_edit+'¥n'

#次の解析用のデータの出力
f=open(file_output,'w')
f.write(new_inp)
f.close
```

副プログラム

#初期剛性

##卍型試験体の場合

```
for i in range(int(data_inp[1][1])):
    if float(data_inp[x+i][3])==0.0 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.5:
        load_1=data_inp[x+i][1]
    elif float(data_inp[x+i][3])==0.375 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.5:
        load_2=data_inp[x+i][1]
    elif float(data_inp[x+i][3])==0.750 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.5:
        load_3=data_inp[x+i][1]
    elif float(data_inp[x+i][3])==1.125 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.5:
        load_4=data_inp[x+i][1]
disp=(float(data_otp[y+int(load_1)-
1001][1])+float(data_otp[y+int(load_4)-1001][1]))/2
load=float(data_inp[x+int(load_1)-
1001][14])+float(data_inp[x+int(load_2)-
1001][14])+float(data_inp[x+int(load_3)-
1001][14])+float(data_inp[x+int(load_4)-1001][14])
katamuki=load*9.8/(disp*1000)
```

##小型試験体用

```
for i in range(int(data_inp[1][1])):
    if float(data_inp[x+i][3])==0.0 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.0:
        load_1=data_inp[x+i][1]
    elif float(data_inp[x+i][3])==0.375 and
float(data_inp[x+i][4])==0.0 and float(data_inp[x+i][5])==1.0:
        load_2=data_inp[x+i][1]
disp=(float(data_otp[y+int(load_1)-
1001][1])+float(data_otp[y+int(load_2)-1001][1]))/2
load=float(data_inp[x+int(load_1)-
1001][14])+float(data_inp[x+int(load_2)-1001][14])
katamuki=load*9.8/(disp*1000)
```

#錫ばね位置表示

```
import matplotlib.pyplot as plt
data_initial=[]
for line in open(file_name+str(1)+'.inp',"r"):
    read_file=line.split()
    data_initial.append(read_file)
open(file_name+str(1)+'.inp',"r").close
data_inp_new=[]
for line in open(file_output, "r"):
    read_inp_new=line.split()
    data_inp_new.append(read_inp_new)
open(file_output, "r").close
tin_x=[]
tin_z=[]
tin=[]
for i in range(int(data_inp_new[2][1])):
    if int(data_inp_new[i+z][3])==404 or
int(data_inp_new[i+z][3])==504:

tin_x.append(float(data_initial[int(data_initial[int(data_inp_n
ew[i+z][1])+z-1001][7])+x-1001][3]))

tin_z.append(float(data_initial[int(data_initial[int(data_inp_n
ew[i+z][1])+z-1001][7])+x-1001][5]))
        tin.append(int(data_inp_new[i+z][1]))
plt.figure(figsize=(11.25,15))
plt.scatter(tin_x,tin_z,color="red")
plt.xlim(0,1.125)
plt.ylim(0,1.5)
plt.plot([0,1.125,1.125,0,0],[0,0,1.5,1.5,0],color="black")
plt.plot([0,0.75],[0.5,0.5],color="black")
plt.plot([0.375,1.125],[1.0,1.0],color="black")
plt.plot([0.375,0.375],[0.5,1.5],color="black")
plt.plot([0.75,0.75],[0.0,1.0],color="black")
plt.axis("off")
plt.show()
```

2.錫ばねの剛性変化判定プログラム

本論文における増分解析を行う際に 1 つ前の入力データと解析結果のデータをもとに、次の解析の入力データを作成するプログラムである。前処理プログラムでは 3DCAD ソフトの Rhinoceros で作成した解析モデルを Rhinoceros のプラグインである Grasshopper を用いて出力したテキストファイルのデータの整型を行った。具体的には、節点条件や荷重条件の編集と主プログラムにおいて端点を共有する錫ばねの中から最も短いものを選択できるように端点を共有する錫ばねをまとめておくという操作を行った。また、副プログラムでは荷重変形曲線の表示や、剛性を発揮している錫ばねの位置及びその剛性の大小の色表示を行うプログラムである。

前処理プログラム

```
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')
data=[]

for line in open("manji12mm_z_6002.inp", "r"):
    read_data=line.split()
    data_edit_1=map(str,read_data)
    data_edit_2=" ".join(data_edit_1)
    data_edit_3=data_edit_2.split()
    data.append(data_edit_3)

x=16+5*(int(data[3][1]))+1+15*(int(data[4][1]))+1
y=x+int(data[1][1])+1

#節点条件
for i in range(int(data[1][1])):
    data[x+i][3]="{0:.9f}".format(float(data[x+i][3]))
    data[x+i][4]="{0:.9f}".format(float(data[x+i][4]))
    data[x+i][5]="{0:.9f}".format(float(data[x+i][5]))

    if data[x+i][3]=='0.000000000' and data[x+i][5]=='0.000000000':
        data[x+i][7]='1'
        data[x+i][8]='1'
        data[x+i][9]='1'
        data[x+i][10]='1'
        data[x+i][11]='1'
        data[x+i][12]='1'
    elif data[x+i][3]=='0.375000000' and data[x+i][5]=='0.000000000':
        data[x+i][7]='1'
        data[x+i][8]='1'
        data[x+i][9]='1'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'
    elif data[x+i][3]=='0.750000000' and data[x+i][5]=='0.000000000':
        data[x+i][7]='1'
        data[x+i][8]='1'
        data[x+i][9]='1'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'
    elif data[x+i][3]=='1.125000000' and data[x+i][5]=='0.000000000':
        data[x+i][7]='1'
        data[x+i][8]='1'
        data[x+i][9]='1'
        data[x+i][10]='1'
        data[x+i][11]='1'
        data[x+i][12]='1'
    elif data[x+i][3]=='0.000000000' and data[x+i][5]=='1.500000000':
        data[x+i][7]='0'
        data[x+i][8]='1'
        data[x+i][9]='0'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'
    elif data[x+i][3]=='0.375000000' and data[x+i][5]=='1.500000000':
        data[x+i][7]='0'
        data[x+i][8]='1'
        data[x+i][9]='0'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'
    elif data[x+i][3]=='0.750000000' and data[x+i][5]=='1.500000000':
        data[x+i][7]='0'
        data[x+i][8]='1'
        data[x+i][9]='0'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'
    elif data[x+i][3]=='1.125000000' and data[x+i][5]=='1.500000000':
        data[x+i][7]='0'
        data[x+i][8]='1'
        data[x+i][9]='0'
        data[x+i][10]='1'
        data[x+i][11]='0'
        data[x+i][12]='1'

    data[x+i][3]="{0:.9f}".format(float(data[x+i][3]))
    data[x+i][4]="{0:.9f}".format(float(data[x+i][4]))
    data[x+i][5]="{0:.9f}".format(float(data[x+i][5]))

#荷重の位置と大きさの設定
for i in range(int(data[1][1])):
    if data[x+i][3]=='0.000000000' and data[x+i][4]=='0.000000000' and data[x+i][5]=='1.500000000':
        data[x+i][14]='0.00001'
    elif data[x+i][3]=='0.375000000' and data[x+i][4]=='0.000000000' and data[x+i][5]=='1.500000000':
        data[x+i][14]='0.00001'
    elif data[x+i][3]=='0.750000000' and data[x+i][4]=='0.000000000' and data[x+i][5]=='1.500000000':
        data[x+i][14]='0.00001'
    elif data[x+i][3]=='1.125000000' and data[x+i][4]=='0.000000000' and data[x+i][5]=='1.500000000':
        data[x+i][14]='0.00001'

#錫ばねをガラス側の端点を共有する錫ばねでグループ化し、グ
```

ループごとに順番に並べ替える

```
tin_list=[]
for i in range(int(data[2][1])):
    if 400<int(data[y+i][3])<800:
        tin_list.append(data[y+i])

tin_elem_num=[]
for i in range(len(tin_list)):
    tin_elem_num.append(tin_list[i][1])
```

#Rhino 上で錫ばね材を上下左右分けたレイヤーで作成しておくことで、角部分の錫ばね材が同一グループにまとめられてしまうことを防ぐ

```
sortsecond_1=lambda val : val[3]
tin_list_sort_1=sorted(tin_list,key=sortsecond_1)

sortsecond_2=lambda val : val[8]
tin_401=[]
tin_411=[]
tin_501=[]
tin_511=[]
tin_601=[]
tin_611=[]
tin_701=[]
tin_711=[]

for i in range(len(tin_list_sort_1)):
    if int(tin_list_sort_1[i][3])==401:
        tin_401.append(tin_list_sort_1[i])
        tin_401_2=sorted(tin_401,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==411:
        tin_411.append(tin_list_sort_1[i])
        tin_411_2=sorted(tin_411,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==501:
        tin_501.append(tin_list_sort_1[i])
        tin_501_2=sorted(tin_501,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==511:
        tin_511.append(tin_list_sort_1[i])
        tin_511_2=sorted(tin_511,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==601:
        tin_601.append(tin_list_sort_1[i])
        tin_601_2=sorted(tin_601,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==611:
        tin_611.append(tin_list_sort_1[i])
        tin_611_2=sorted(tin_611,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==701:
        tin_701.append(tin_list_sort_1[i])
        tin_701_2=sorted(tin_701,key=sortsecond_2)
    elif int(tin_list_sort_1[i][3])==711:
        tin_711.append(tin_list_sort_1[i])
        tin_711_2=sorted(tin_711,key=sortsecond_2)

tin_list_new=[]
for j in range(len(tin_401_2)):
    tin_list_new.append(tin_401_2[j])
for j in range(len(tin_501_2)):
    tin_list_new.append(tin_501_2[j])
for j in range(len(tin_601_2)):
    tin_list_new.append(tin_601_2[j])
for j in range(len(tin_701_2)):
    tin_list_new.append(tin_701_2[j])
for j in range(len(tin_411_2)):
    tin_list_new.append(tin_411_2[j])
for j in range(len(tin_511_2)):
    tin_list_new.append(tin_511_2[j])
for j in range(len(tin_611_2)):
    tin_list_new.append(tin_611_2[j])
for j in range(len(tin_711_2)):
    tin_list_new.append(tin_711_2[j])

for i in range(len(tin_list_new)):
    tin_list_new[i][1]=tin_elem_num[i]
```

#端の錫ばねと間の錫ばねで別の SECT に振り分ける

```
for i in range(len(tin_list_new)):
    if int(tin_list_new[i][3])%100==1:
        tin_list_new[i][3]='401'
    elif int(tin_list_new[i][3])%100==11:
        tin_list_new[i][3]='501'

for i in range(len(tin_list_new)):
    data[i+y+int(tin_list_new[0][1])-1001]=tin_list_new[i]
for i in range(len(data)):
    data[i]=" ".join(data[i])
data_1=data
data_2='¥n'.join(data_1)
f=open('manji12mm_z_6005_1.inp','w')
f.write(data_2)
f.close
```

主プログラム

```
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')

import numpy as np
import math

#ファイル名、ステップ数、錫ばねの長さ
file_name='manji12mm_z_6014_'
step=33
tin=0.001

next_step=step+1
file_input_1=file_name+str(step)+'.inp'
file_input_2=file_name+str(step)+'.otp'
file_output=file_name+str(next_step)+'.inp'

#データの読み込み
data_inp=[]
data_otp=[]
data_initial=[]


#解析モデルの入力ファイル
for line in open(file_input_1, "r"):
    read_inp=line.split()
    data_inp_edit_1=map(str,read_inp)
    data_inp_edit_2=" ".join(data_inp_edit_1)
    data_inp_edit_3=data_inp_edit_2.split()
    data_inp.append(data_inp_edit_3)
open(file_input_1, "r").close
#解析モデルの出力ファイル
for line in open(file_input_2, "r"):
    read_otp=line.split()
    data_otp_edit_1=map(str,read_otp)
    data_otp_edit_2=" ".join(data_otp_edit_1)
    data_otp_edit_3=data_otp_edit_2.split()
    data_otp.append(data_otp_edit_3)
open(file_input_2, "r").close
#解析モデルの初期データ(step1)
for line in open(file_name+str(1)+'.inp',"r"):
    read_file=line.split()
    data_initial.append(read_file)
open(file_name+str(1)+'.inp',"r").close

x=15+int(data_inp[3][1])*5+1+(int(data_inp[4][1]))*15+2
y=6+int(data_inp[2][1])*2+6
z=x+int(data_inp[1][1])+1


#inp ファイルの点の座標の編集
```

```
for i in range(int(data_inp[1][1])):
    for j in range(3):

data_inp[x+i][3+j]="{0:.9f}".format(float(data_inp[x+i][3+j])+fl
oat(data_otp[y+i][1+j]))

#inp ファイルの CMQ の編集
for i in range(int(data_inp[2][1])):
    for j in range(6):
        data_inp[z+i][j+25]="{0:.9f}".format(-
float(data_otp[6+2*i][3+j]))
        data_inp[z+i][j+31]="{0:.9f}".format(-
float(data_otp[7+2*i][1+j]))

#単点を共有する錫ばねの長さ比較し、最も短い錫ばね材に剛性
判定をおこない、他の錫ばね材は全て剛性を与えない
##錫ばねのみ取り出したリストを作成する
tin_list=[]
tin_length=[]

for i in range(int(data_inp[2][1])):
    if 400<int(data_inp[z+i][3])<600:
        tin_list.append(data_inp[z+i])
        tin_length.append(math.sqrt((float(data_inp[int(data
        _inp[z+i][7])-(1001-x)][3])-
        float(data_inp[int(data_inp[z+i][8])-(1001-
        x)][3]))**2+(float(data_inp[int(data_inp[z+i][7])-
        (1001-x)][5])-float(data_inp[int(data_inp[z+i][8])-
        (1001-x)][5]))**2))

##単点ごとに錫をグループ化
group_tin=[]

k=1
group_tin.append(1)
for i in range(len(tin_list)-1):
    if int(tin_list[i][8])==int(tin_list[i+1][8]):
        group_tin.append(k)
    else:
        group_tin.append(k+1)
        k=k+1

##1 グループの大きさ
l=1
group_size_tin=[]
for i in range(len(group_tin)-1):
    if i<len(group_tin)-2 and group_tin[i]==group_tin[i+1]:
        l=l+1
    elif i==len(group_tin)-2:
        l=l+1
        group_size_tin.append(l)
    else:
        group_size_tin.append(l)
        l=1
##グループ内で何番目の錫ばねなのか
tin_num=[]
p=0
tin_num.append(p)
for i in range(len(group_size_tin)-1):
    p=p+group_size_tin[i]
    tin_num.append(p)

##グループごとに錫ばねの長さを比較し、1mm 以下の錫ばねを
取り出す
l_min_list=[]
l_min_elem=[]

for i in range(len(group_size_tin)):
    l_min=10
    for j in range(group_size_tin[i]):
        if l_min>=tin_length[j+int(tin_num[i])]:
            l_min=tin_length[j+int(tin_num[i])]
            l_min_tin_num=tin_list[j+int(tin_num[i])][1]
    if l_min<=tin:
        l_min_list.append(l_min)
        l_min_elem.append(l_min_tin_num)

##錫ばねをすべて Air_Tin に置き換える
for i in range(int(data_inp[2][1])):
    if int(data_inp[i+z][3])//100==4:
        data_inp[i+z][3]='401'
    elif int(data_inp[i+z][3])//100==5:
        data_inp[i+z][3]='501'

##l_min_elem の縮み調べ、大きさに応じた剛性の錫ばねを入力
for i in range(len(l_min_elem)):
    if int(data_inp[z+int(l_min_elem[i])-1001][3])==401:
        if l_min_list[i]>=tin-0.0000862:
            data_inp[z+int(l_min_elem[i])-1001][3]='402'
        elif tin-0.0001126<=l_min_list[i]<tin-0.0000862:
            data_inp[z+int(l_min_elem[i])-1001][3]='403'
        elif l_min_list[i]<tin-0.0001126:
            data_inp[z+int(l_min_elem[i])-1001][3]='404'
    if int(data_inp[z+int(l_min_elem[i])-1001][3])==501:
        if l_min_list[i]>=tin-0.0000862:
            data_inp[z+int(l_min_elem[i])-1001][3]='502'
        elif tin-0.0001126<=l_min_list[i]<tin-0.0000862:
            data_inp[z+int(l_min_elem[i])-1001][3]='503'
        elif l_min_list[i]<tin-0.0001126:
            data_inp[z+int(l_min_elem[i])-1001][3]='504'


#荷重の変更
if step==10:
    new_load='0.00002'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==15:
    new_load='0.00005'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==20:
    new_load='0.00015'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==25:
    new_load='0.00025'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==30:
    new_load='0.00050'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==35:
    new_load='0.00100'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==40:
    new_load='0.00150'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==45:
    new_load='0.00250'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==50:
    new_load='0.00400'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==60:
    new_load='0.00600'
```

```
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==70:
    new_load='0.00800'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==80:
    new_load='0.01000'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==90:
    new_load='0.02000'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==100:
    new_load='0.05000'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==110:
    new_load='0.08000'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load
if step==120:
    new_load='0.10000'
    for i in range(int(data_initial[1][1])):
        if float(data_initial[x+i][14])!=0.0:
            data_inp[x+i][14]=new_load

#突き抜けチェック(ガラスが鉄骨を突き抜けてしまう可能性の
ある部分)
for i in range(int(data_initial[1][1])):
    if      float(data_initial[x+i][3])==0.0      and
float(data_initial[x+i][5])==1.5:
        steel_edge=x+i
    elif    float(data_initial[x+i][3])==0.00225    and
float(data_initial[x+i][5])==1.5-(0.0045-tin):
        glass_edge=x+i

#新しいファイルの書き出し
data_inp_edit=[]
for i in range(len(data_inp)):
    data_inp_edit.append(data_inp[i])
for i in range(len(data_inp_edit)):
    data_inp_edit[i]=" ".join(data_inp_edit[i])
data_inp_new='¥n'.join(data_inp_edit)+'¥n'
f=open(file_output,'w')
f.write(data_inp_new)
f.close
```

副プログラム

```
#錫ばねの位置表示
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')

import matplotlib.pyplot as plt

#解析モデルの初期データの読み込み
data_initial=[]
for line in open(file_name+str(1)+'.inp',"r"):
    read_file=line.split()
    data_initial.append(read_file)
open(file_name+str(1)+'.inp',"r").close

#剛性の異なる錫ばねを分けて読み込む
tin_aida_k1=[]
tin_aida_k2=[]
tin_aida_k3=[]
tin_hashi_k1=[]
tin_hashi_k2=[]
tin_hashi_k3=[]

tin_aida_k1_x=[]
tin_aida_k1_z=[]
tin_aida_k2_x=[]
tin_aida_k2_z=[]
tin_aida_k3_x=[]
tin_aida_k3_z=[]
tin_hashi_k1_x=[]
tin_hashi_k1_z=[]
tin_hashi_k2_x=[]
tin_hashi_k2_z=[]
tin_hashi_k3_x=[]
tin_hashi_k3_z=[]

#次の解析における解析モデルのデータ上で発生している錫ばね材の位置
data_tin_checker=[]
for line in open(file_name+str(step+1)+'.inp',"r"):
    read_file=line.split()
    data_tin_checker.append(read_file)
open(file_name+str(step+1)+'.inp',"r").close

#錫ばね材のガラスと端点を共有している側の節点の座標を読み取る
for i in range(int(data_tin_checker[2][1])):
    if int(data_tin_checker[i+z][3])==402:

tin_aida_k1_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_aida_k1_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_aida_k1.append(int(data_tin_checker[i+z][1]))
    elif int(data_tin_checker[i+z][3])==403:

tin_aida_k2_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_aida_k2_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_aida_k2.append(int(data_tin_checker[i+z][1]))
    elif int(data_tin_checker[i+z][3])==404:

tin_aida_k3_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_aida_k3_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_aida_k3.append(int(data_tin_checker[i+z][1]))
    elif int(data_tin_checker[i+z][3])==502:

tin_hashi_k1_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_hashi_k1_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_hashi_k1.append(int(data_tin_checker[i+z][1]))
    elif int(data_tin_checker[i+z][3])==503:

tin_hashi_k2_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_hashi_k2_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_hashi_k2.append(int(data_tin_checker[i+z][1]))
    elif int(data_tin_checker[i+z][3])==504:

tin_hashi_k3_x.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][3]))

tin_hashi_k3_z.append(float(data_initial[int(data_initial[int(data_tin_checker[i+z][1])+z-1001][8])+x-1001][5]))
        tin_hashi_k3.append(int(data_tin_checker[i+z][1]))


plt.figure(figsize=(5.625,7.5))
plt.scatter(tin_aida_k1_x,tin_aida_k1_z,color="cyan")
plt.scatter(tin_aida_k2_x,tin_aida_k2_z,color="deepskyblue")
plt.scatter(tin_aida_k3_x,tin_aida_k3_z,color="blue")
plt.scatter(tin_hashi_k1_x,tin_hashi_k1_z,color="pink")
plt.scatter(tin_hashi_k2_x,tin_hashi_k2_z,color="hotpink")
plt.scatter(tin_hashi_k3_x,tin_hashi_k3_z,color="deeppink")

plt.xlim(0,1.125)
plt.ylim(0,1.5)
plt.plot([0,1.125,1.125,0,0],[0,0,1.5,1.5,0],color="black")
plt.plot([0,0.75],[0.5,0.5],color="black")
plt.plot([0.375,1.125],[1.0,1.0],color="black")
plt.plot([0.375,0.375],[0.5,1.5],color="black")
plt.plot([0.75,0.75],[0.0,1.0],color="black")
plt.axis("off")

plt.show()


#荷重変形曲線(for のループで全ての解析の inp と otp から荷重と変位を読み込むので step 数増えてくると時間がかかる。解析がやり直しすることなく回しきれているうちは新しい荷重と変位のデータを追加していくプログラムの方が早いが、解析がスムーズに進まずある step で荷重を変えてやり直したりしていたため毎回全部読み込む方がミスは起きないのでこのプログラムで行った)
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')

import numpy as np
import math

##変位を測定する位置
for i in range(int(data_initial[1][1])):
    if float(data_initial[x+i][3])==0.0 and float(data_initial[x+i][4])==0.0 and float(data_initial[x+i][5])==1.5:
        left=x+i
    elif float(data_initial[x+i][3])==1.125 and float(data_initial[x+i][4])==0.0 and float(data_initial[x+i][5])==1.5:
        right=x+i

##荷重を加えている位置データ
for i in range(int(data_initial[1][1])):
```

```
    if              float(data_initial[x+i][3])==0.0            and
float(data_initial[x+i][4])==0.0                                and
float(data_initial[x+i][5])==1.5:
        load_1=x+i
    elif            float(data_initial[x+i][3])==0.375          and
float(data_initial[x+i][4])==0.0                                and
float(data_initial[x+i][5])==1.5:
        load_2=x+i
    elif             float(data_initial[x+i][3])==0.75           and
float(data_initial[x+i][4])==0.0                                and
float(data_initial[x+i][5])==1.5:
        load_3=x+i
    elif            float(data_initial[x+i][3])==1.125           and
float(data_initial[x+i][4])==0.0                                and
float(data_initial[x+i][5])==1.5:
        load_4=x+i

for i in range(step+2):
    if i==0:
        disp=[]
        load_sum=[]
        load=0
        load_sum.append(load)
    elif i==step+1:
        data_1=[]
        for line in open(file_name+str(i)+'.inp',"r"):
            read_inp=line.split()
            read_inp_1=map(str,read_inp)
            read_inp_2=" ".join(read_inp_1)
            read_inp_3=read_inp_2.split()
            data_1.append(read_inp_3)

        open(file_name+str(i)+'.inp',"r").close

        disp1=(float(data_1[left][3]))*1000
        disp2=(float(data_1[right][3])-1.125)*1000
        disp.append((disp1+disp2)/2)
    else:
        data_1=[]
        for line in open(file_name+str(i)+'.inp',"r"):
            read_inp=line.split()
            read_inp_1=map(str,read_inp)
            read_inp_2=" ".join(read_inp_1)
            read_inp_3=read_inp_2.split()
            data_1.append(read_inp_3)

        open(file_name+str(i)+'.inp',"r").close

        disp1=(float(data_1[left][3]))*1000
        disp2=(float(data_1[right][3])-1.125)*1000
        disp.append((disp1+disp2)/2)
        load=load+(float(data_1[left][14]))*9.8+(float(data_1[l
        oad_2][14]))*9.8+(float(data_1[load_3][14]))*9.8+(floa
        t(data_1[load_4][14]))*9.8
        load_sum.append(load)

#剛性値
katamuki=[]

for i in range(len(disp)-1):
    katamuki.append((float(load_sum[i+1])-
float(load_sum[i]))/(float(disp[i+1])-float(disp[i])))

x=disp
y=load_sum

plt.plot(x,y)
plt.show()

#個別のガラスごとの錫ばねの位置表示
glass1_tin_aida_k1=[]
glass1_tin_hashi_k1=[]
glass1_tin_aida_k1_x=[]
glass1_tin_aida_k1_z=[]
glass1_tin_hashi_k1_x=[]
glass1_tin_hashi_k1_z=[]
glass1_tin_aida_k2=[]
glass1_tin_hashi_k2=[]
glass1_tin_aida_k2_x=[]
glass1_tin_aida_k2_z=[]
glass1_tin_hashi_k2_x=[]
glass1_tin_hashi_k2_z=[]
glass1_tin_aida_k3=[]
glass1_tin_hashi_k3=[]
glass1_tin_aida_k3_x=[]
glass1_tin_aida_k3_z=[]
glass1_tin_hashi_k3_x=[]
glass1_tin_hashi_k3_z=[]

glass2_tin_aida_k1=[]
glass2_tin_hashi_k1=[]
glass2_tin_aida_k1_x=[]
glass2_tin_aida_k1_z=[]
glass2_tin_hashi_k1_x=[]
glass2_tin_hashi_k1_z=[]
glass2_tin_aida_k2=[]
glass2_tin_hashi_k2=[]
glass2_tin_aida_k2_x=[]
glass2_tin_aida_k2_z=[]
glass2_tin_hashi_k2_x=[]
glass2_tin_hashi_k2_z=[]
glass2_tin_aida_k3=[]
glass2_tin_hashi_k3=[]
glass2_tin_aida_k3_x=[]
glass2_tin_aida_k3_z=[]
glass2_tin_hashi_k3_x=[]
glass2_tin_hashi_k3_z=[]

glass3_tin_aida_k1=[]
glass3_tin_hashi_k1=[]
glass3_tin_aida_k1_x=[]
glass3_tin_aida_k1_z=[]
glass3_tin_hashi_k1_x=[]
glass3_tin_hashi_k1_z=[]
glass3_tin_aida_k2=[]
glass3_tin_hashi_k2=[]
glass3_tin_aida_k2_x=[]
glass3_tin_aida_k2_z=[]
glass3_tin_hashi_k2_x=[]
glass3_tin_hashi_k2_z=[]
glass3_tin_aida_k3=[]
glass3_tin_hashi_k3=[]
glass3_tin_aida_k3_x=[]
glass3_tin_aida_k3_z=[]
glass3_tin_hashi_k3_x=[]
glass3_tin_hashi_k3_z=[]

glass4_tin_aida_k1=[]
glass4_tin_hashi_k1=[]
glass4_tin_aida_k1_x=[]
glass4_tin_aida_k1_z=[]
glass4_tin_hashi_k1_x=[]
glass4_tin_hashi_k1_z=[]
glass4_tin_aida_k2=[]
glass4_tin_hashi_k2=[]
glass4_tin_aida_k2_x=[]
glass4_tin_aida_k2_z=[]
glass4_tin_hashi_k2_x=[]
glass4_tin_hashi_k2_z=[]
glass4_tin_aida_k3=[]
glass4_tin_hashi_k3=[]
glass4_tin_aida_k3_x=[]
glass4_tin_aida_k3_z=[]
glass4_tin_hashi_k3_x=[]
glass4_tin_hashi_k3_z=[]

glass5_tin_aida_k1=[]
glass5_tin_hashi_k1=[]
glass5_tin_aida_k1_x=[]
glass5_tin_aida_k1_z=[]
glass5_tin_hashi_k1_x=[]
glass5_tin_hashi_k1_z=[]
glass5_tin_aida_k2=[]
glass5_tin_hashi_k2=[]
glass5_tin_aida_k2_x=[]
```

```
glass5_tin_aida_k2_z=[]
glass5_tin_hashi_k2_x=[]
glass5_tin_hashi_k2_z=[]
glass5_tin_aida_k3=[]
glass5_tin_hashi_k3=[]
glass5_tin_aida_k3_x=[]
glass5_tin_aida_k3_z=[]
glass5_tin_hashi_k3_x=[]
glass5_tin_hashi_k3_z=[]

##glass1
for i in range(len(tin_aida_k1_x)):
    if 0.0<tin_aida_k1_x[i]<0.375 and 0.5<tin_aida_k1_z[i]<1.5:
        glass1_tin_aida_k1_x.append(tin_aida_k1_x[i])
        glass1_tin_aida_k1_z.append(tin_aida_k1_z[i])
        glass1_tin_aida_k1.append(tin_aida_k1[i])
for i in range(len(tin_hashi_k1_x)):
    if 0.0<tin_hashi_k1_x[i]<0.375 and 0.5<tin_hashi_k1_z[i]<1.5:
        glass1_tin_hashi_k1_x.append(tin_hashi_k1_x[i])
        glass1_tin_hashi_k1_z.append(tin_hashi_k1_z[i])
        glass1_tin_hashi_k1.append(tin_hashi_k1[i])
for i in range(len(tin_aida_k2_x)):
    if 0.0<tin_aida_k2_x[i]<0.375 and 0.5<tin_aida_k2_z[i]<1.5:
        glass1_tin_aida_k2_x.append(tin_aida_k2_x[i])
        glass1_tin_aida_k2_z.append(tin_aida_k2_z[i])
        glass1_tin_aida_k2.append(tin_aida_k2[i])
for i in range(len(tin_hashi_k2_x)):
    if 0.0<tin_hashi_k2_x[i]<0.375 and 0.5<tin_hashi_k2_z[i]<1.5:
        glass1_tin_hashi_k2_x.append(tin_hashi_k2_x[i])
        glass1_tin_hashi_k2_z.append(tin_hashi_k2_z[i])
        glass1_tin_hashi_k2.append(tin_hashi_k2[i])
for i in range(len(tin_aida_k3_x)):
    if 0.0<tin_aida_k3_x[i]<0.375 and 0.5<tin_aida_k3_z[i]<1.5:
        glass1_tin_aida_k3_x.append(tin_aida_k3_x[i])
        glass1_tin_aida_k3_z.append(tin_aida_k3_z[i])
        glass1_tin_aida_k3.append(tin_aida_k3[i])
for i in range(len(tin_hashi_k3_x)):
    if 0.0<tin_hashi_k3_x[i]<0.375 and 0.5<tin_hashi_k3_z[i]<1.5:
        glass1_tin_hashi_k3_x.append(tin_hashi_k3_x[i])
        glass1_tin_hashi_k3_z.append(tin_hashi_k3_z[i])
        glass1_tin_hashi_k3.append(tin_hashi_k3[i])

plt.figure(figsize=(1.5,4))
plt.scatter(glass1_tin_aida_k1_x,glass1_tin_aida_k1_z,color="cyan")
plt.scatter(glass1_tin_aida_k2_x,glass1_tin_aida_k2_z,color="deepskyblue")
plt.scatter(glass1_tin_aida_k3_x,glass1_tin_aida_k3_z,color="blue")
plt.scatter(glass1_tin_hashi_k1_x,glass1_tin_hashi_k1_z,color="pink")
plt.scatter(glass1_tin_hashi_k2_x,glass1_tin_hashi_k2_z,color="hotpink")
plt.scatter(glass1_tin_hashi_k3_x,glass1_tin_hashi_k3_z,color="deeppink")

plt.xlim(0.0,0.375)
plt.ylim(0.5,1.5)
plt.plot([0,0.375,0.375,0.0,0.0],[0.5,0.5,1.5,1.5,0.5],color="black")
plt.axis("off")
plt.show()

##glass2
for i in range(len(tin_aida_k1_x)):
    if 0.375<tin_aida_k1_x[i]<1.125 and 1.0<tin_aida_k1_z[i]<1.5:
        glass2_tin_aida_k1_x.append(tin_aida_k1_x[i])
        glass2_tin_aida_k1_z.append(tin_aida_k1_z[i])
        glass2_tin_aida_k1.append(tin_aida_k1[i])
for i in range(len(tin_hashi_k1_x)):
    if 0.375<tin_hashi_k1_x[i]<1.125 and 1.0<tin_hashi_k1_z[i]<1.5:
        glass2_tin_hashi_k1_x.append(tin_hashi_k1_x[i])
        glass2_tin_hashi_k1_z.append(tin_hashi_k1_z[i])
        glass2_tin_hashi_k1.append(tin_hashi_k1[i])
for i in range(len(tin_aida_k2_x)):
    if 0.375<tin_aida_k2_x[i]<1.125 and 1.0<tin_aida_k2_z[i]<1.5:
        glass2_tin_aida_k2_x.append(tin_aida_k2_x[i])
        glass2_tin_aida_k2_z.append(tin_aida_k2_z[i])
        glass2_tin_aida_k2.append(tin_aida_k2[i])
for i in range(len(tin_hashi_k2_x)):
    if 0.375<tin_hashi_k2_x[i]<1.125 and 1.0<tin_hashi_k2_z[i]<1.5:
        glass2_tin_hashi_k2_x.append(tin_hashi_k2_x[i])
        glass2_tin_hashi_k2_z.append(tin_hashi_k2_z[i])
        glass2_tin_hashi_k2.append(tin_hashi_k2[i])
for i in range(len(tin_aida_k3_x)):
    if 0.375<tin_aida_k3_x[i]<1.125 and 1.0<tin_aida_k3_z[i]<1.5:
        glass2_tin_aida_k3_x.append(tin_aida_k3_x[i])
        glass2_tin_aida_k3_z.append(tin_aida_k3_z[i])
        glass2_tin_aida_k3.append(tin_aida_k3[i])
for i in range(len(tin_hashi_k3_x)):
    if 0.375<tin_hashi_k3_x[i]<1.125 and 1.0<tin_hashi_k3_z[i]<1.5:
        glass2_tin_hashi_k3_x.append(tin_hashi_k3_x[i])
        glass2_tin_hashi_k3_z.append(tin_hashi_k3_z[i])
        glass2_tin_hashi_k3.append(tin_hashi_k3[i])

plt.figure(figsize=(3,2))
plt.scatter(glass2_tin_aida_k1_x,glass2_tin_aida_k1_z,color="cyan")
plt.scatter(glass2_tin_aida_k2_x,glass2_tin_aida_k2_z,color="deepskyblue")
plt.scatter(glass2_tin_aida_k3_x,glass2_tin_aida_k3_z,color="blue")
plt.scatter(glass2_tin_hashi_k1_x,glass2_tin_hashi_k1_z,color="pink")
plt.scatter(glass2_tin_hashi_k2_x,glass2_tin_hashi_k2_z,color="hotpink")
plt.scatter(glass2_tin_hashi_k3_x,glass2_tin_hashi_k3_z,color="deeppink")

plt.xlim(0.375,1.125)
plt.ylim(1.0,1.5)
plt.plot([0.375,1.125,1.125,0.375,0.375],[1.0,1.0,1.5,1.5,1.0],color="black")
plt.axis("off")
plt.show()

##glass3
for i in range(len(tin_aida_k1_x)):
    if 0.75<tin_aida_k1_x[i]<1.125 and 0.0<tin_aida_k1_z[i]<1.0:
        glass3_tin_aida_k1_x.append(tin_aida_k1_x[i])
        glass3_tin_aida_k1_z.append(tin_aida_k1_z[i])
        glass3_tin_aida_k1.append(tin_aida_k1[i])
for i in range(len(tin_hashi_k1_x)):
    if 0.75<tin_hashi_k1_x[i]<1.125 and 0.0<tin_hashi_k1_z[i]<1.0:
        glass3_tin_hashi_k1_x.append(tin_hashi_k1_x[i])
        glass3_tin_hashi_k1_z.append(tin_hashi_k1_z[i])
        glass3_tin_hashi_k1.append(tin_hashi_k1[i])
for i in range(len(tin_aida_k2_x)):
    if 0.75<tin_aida_k2_x[i]<1.125 and 0.0<tin_aida_k2_z[i]<1.0:
        glass3_tin_aida_k2_x.append(tin_aida_k2_x[i])
        glass3_tin_aida_k2_z.append(tin_aida_k2_z[i])
        glass3_tin_aida_k2.append(tin_aida_k2[i])
for i in range(len(tin_hashi_k2_x)):
    if 0.75<tin_hashi_k2_x[i]<1.125 and 0.0<tin_hashi_k2_z[i]<1.0:
        glass3_tin_hashi_k2_x.append(tin_hashi_k2_x[i])
        glass3_tin_hashi_k2_z.append(tin_hashi_k2_z[i])
        glass3_tin_hashi_k2.append(tin_hashi_k2[i])
for i in range(len(tin_aida_k3_x)):
    if 0.75<tin_aida_k3_x[i]<1.125 and 0.0<tin_aida_k3_z[i]<1.0:
```

```
            glass3_tin_aida_k3_x.append(tin_aida_k3_x[i])
            glass3_tin_aida_k3_z.append(tin_aida_k3_z[i])
            glass3_tin_aida_k3.append(tin_aida_k3[i])
for i in range(len(tin_hashi_k3_x)):
    if 0.75<tin_hashi_k3_x[i]<1.125 and 0.0<tin_hashi_k3_z[i]<1.0:
            glass3_tin_hashi_k3_x.append(tin_hashi_k3_x[i])
            glass3_tin_hashi_k3_z.append(tin_hashi_k3_z[i])
            glass3_tin_hashi_k3.append(tin_hashi_k3[i])

plt.figure(figsize=(1.5,4))
plt.scatter(glass3_tin_aida_k1_x,glass3_tin_aida_k1_z,color="cyan")
plt.scatter(glass3_tin_aida_k2_x,glass3_tin_aida_k2_z,color="deepskyblue")
plt.scatter(glass3_tin_aida_k3_x,glass3_tin_aida_k3_z,color="blue")
plt.scatter(glass3_tin_hashi_k1_x,glass3_tin_hashi_k1_z,color="pink")
plt.scatter(glass3_tin_hashi_k2_x,glass3_tin_hashi_k2_z,color="hotpink")
plt.scatter(glass3_tin_hashi_k3_x,glass3_tin_hashi_k3_z,color="deeppink")

plt.xlim(0.75,1.125)
plt.ylim(0.0,1.0)
plt.plot([0.75,1.125,1.125,0.75,0.75],[0.0,0.0,1.0,1.0,0.0],color="black")
plt.axis("off")
plt.show()

##glass4
for i in range(len(tin_aida_k1_x)):
    if 0.0<tin_aida_k1_x[i]<0.75 and 0.0<tin_aida_k1_z[i]<0.5:
            glass4_tin_aida_k1_x.append(tin_aida_k1_x[i])
            glass4_tin_aida_k1_z.append(tin_aida_k1_z[i])
            glass4_tin_aida_k1.append(tin_aida_k1[i])
for i in range(len(tin_hashi_k1_x)):
    if 0.0<tin_hashi_k1_x[i]<0.75 and 0.0<tin_hashi_k1_z[i]<0.5:
            glass4_tin_hashi_k1_x.append(tin_hashi_k1_x[i])
            glass4_tin_hashi_k1_z.append(tin_hashi_k1_z[i])
            glass4_tin_hashi_k1.append(tin_hashi_k1[i])
for i in range(len(tin_aida_k2_x)):
    if 0.0<tin_aida_k2_x[i]<0.75 and 0.0<tin_aida_k2_z[i]<0.5:
            glass4_tin_aida_k2_x.append(tin_aida_k2_x[i])
            glass4_tin_aida_k2_z.append(tin_aida_k2_z[i])
            glass4_tin_aida_k2.append(tin_aida_k2[i])
for i in range(len(tin_hashi_k2_x)):
    if 0.0<tin_hashi_k2_x[i]<0.75 and 0.0<tin_hashi_k2_z[i]<0.5:
            glass4_tin_hashi_k2_x.append(tin_hashi_k2_x[i])
            glass4_tin_hashi_k2_z.append(tin_hashi_k2_z[i])
            glass4_tin_hashi_k2.append(tin_hashi_k2[i])
for i in range(len(tin_aida_k3_x)):
    if 0.0<tin_aida_k3_x[i]<0.75 and 0.0<tin_aida_k3_z[i]<0.5:
            glass4_tin_aida_k3_x.append(tin_aida_k3_x[i])
            glass4_tin_aida_k3_z.append(tin_aida_k3_z[i])
            glass4_tin_aida_k3.append(tin_aida_k3[i])
for i in range(len(tin_hashi_k3_x)):
    if 0.0<tin_hashi_k3_x[i]<0.75 and 0.0<tin_hashi_k3_z[i]<0.5:
            glass4_tin_hashi_k3_x.append(tin_hashi_k3_x[i])
            glass4_tin_hashi_k3_z.append(tin_hashi_k3_z[i])
            glass4_tin_hashi_k3.append(tin_hashi_k3[i])

plt.figure(figsize=(3,2))
plt.scatter(glass4_tin_aida_k1_x,glass4_tin_aida_k1_z,color="cyan")
plt.scatter(glass4_tin_aida_k2_x,glass4_tin_aida_k2_z,color="deepskyblue")
plt.scatter(glass4_tin_aida_k3_x,glass4_tin_aida_k3_z,color="blue")
plt.scatter(glass4_tin_hashi_k1_x,glass4_tin_hashi_k1_z,color="pink")
plt.scatter(glass4_tin_hashi_k2_x,glass4_tin_hashi_k2_z,color="hotpink")
plt.scatter(glass4_tin_hashi_k3_x,glass4_tin_hashi_k3_z,color="deeppink")

plt.xlim(0.0,0.75)
plt.ylim(0.0,0.5)
plt.plot([0,0.75,0.75,0.0,0.0],[0.0,0.0,0.5,0.5,0.0],color="black")
plt.axis("off")
plt.show()

##glass5
for i in range(len(tin_aida_k1_x)):
    if 0.375<tin_aida_k1_x[i]<0.75 and 0.5<tin_aida_k1_z[i]<1.0:
            glass5_tin_aida_k1_x.append(tin_aida_k1_x[i])
            glass5_tin_aida_k1_z.append(tin_aida_k1_z[i])
            glass5_tin_aida_k1.append(tin_aida_k1[i])
for i in range(len(tin_hashi_k1_x)):
    if 0.375<tin_hashi_k1_x[i]<0.75 and 0.5<tin_hashi_k1_z[i]<1.0:
            glass5_tin_hashi_k1_x.append(tin_hashi_k1_x[i])
            glass5_tin_hashi_k1_z.append(tin_hashi_k1_z[i])
            glass5_tin_hashi_k1.append(tin_hashi_k1[i])
for i in range(len(tin_aida_k2_x)):
    if 0.375<tin_aida_k2_x[i]<0.75 and 0.5<tin_aida_k2_z[i]<1.0:
            glass5_tin_aida_k2_x.append(tin_aida_k2_x[i])
            glass5_tin_aida_k2_z.append(tin_aida_k2_z[i])
            glass5_tin_aida_k2.append(tin_aida_k2[i])
for i in range(len(tin_hashi_k2_x)):
    if 0.375<tin_hashi_k2_x[i]<0.75 and 0.5<tin_hashi_k2_z[i]<1.0:
            glass5_tin_hashi_k2_x.append(tin_hashi_k2_x[i])
            glass5_tin_hashi_k2_z.append(tin_hashi_k2_z[i])
            glass5_tin_hashi_k2.append(tin_hashi_k2[i])
for i in range(len(tin_aida_k3_x)):
    if 0.375<tin_aida_k3_x[i]<0.75 and 0.5<tin_aida_k3_z[i]<1.0:
            glass5_tin_aida_k3_x.append(tin_aida_k3_x[i])
            glass5_tin_aida_k3_z.append(tin_aida_k3_z[i])
            glass5_tin_aida_k3.append(tin_aida_k3[i])
for i in range(len(tin_hashi_k3_x)):
    if 0.375<tin_hashi_k3_x[i]<0.75 and 0.5<tin_hashi_k3_z[i]<1.0:
            glass5_tin_hashi_k3_x.append(tin_hashi_k3_x[i])
            glass5_tin_hashi_k3_z.append(tin_hashi_k3_z[i])
            glass5_tin_hashi_k3.append(tin_hashi_k3[i])

plt.figure(figsize=(1.5,2))
plt.scatter(glass5_tin_aida_k1_x,glass5_tin_aida_k1_z,color="cyan")
plt.scatter(glass5_tin_aida_k2_x,glass5_tin_aida_k2_z,color="deepskyblue")
plt.scatter(glass5_tin_aida_k3_x,glass5_tin_aida_k3_z,color="blue")
plt.scatter(glass5_tin_hashi_k1_x,glass5_tin_hashi_k1_z,color="pink")
plt.scatter(glass5_tin_hashi_k2_x,glass5_tin_hashi_k2_z,color="hotpink")
plt.scatter(glass5_tin_hashi_k3_x,glass5_tin_hashi_k3_z,color="deeppink")

plt.xlim(0.375,0.75)
plt.ylim(0.5,1.0)
plt.plot([0.375,0.75,0.75,0.375,0.375],[0.5,0.5,1.0,1.0,0.5],color="black")
plt.axis("off")
plt.show()


#荷重と変形のデータの履歴の出力
disp_edit=map(str,disp)
disp_edit_2=" ".join(disp_edit)
disp_edit_3=disp_edit_2.split()
disp_edit_4='¥n'.join(disp_edit_3)

with open(file_name+'disp.txt', mode='w') as f:
    f.write(disp_edit_4)
```

```
load_sum_edit=map(str,load_sum)
load_sum_edit_2=" ".join(load_sum_edit)
load_sum_edit_3=load_sum_edit_2.split()
load_sum_edit_4='¥n'.join(load_sum_edit_3)

with open(file_name+'load_sum.txt', mode='w') as f:
    f.write(load_sum_edit_4)


#作成したデータでの解析が不安定になった場合の断面 2 次モーメント編集操作
import os
os.chdir('C:¥D¥Cdocs¥Hogan¥Win32¥Debug')

#ファイル、step 数、新しい断面 2 次モーメント
file_name='manji12mm_z_6014_'
step=37
new_i='0.00000004'

file_input=file_name+str(step)+'.inp'
file_output=file_name+str(step)+'.inp'

data_inp=[]
for line in open(file_input, "r"):
    read_inp=line.split()
    data_inp.append(read_inp)
open(file_input, "r").close

x=16+int(data_inp[3][1])*5+1
for i in range(int(data_inp[4][1])):
    if 400<int(data_inp[x+15*i][1])<600:
        data_inp[x+15*i+4][1]=new_i
        data_inp[x+15*i+5][1]=new_i
        data_inp[x+15*i+6][1]=new_i

data_inp_edit=[]
for i in range(len(data_inp)):
    data_inp_edit.append(data_inp[i])
for i in range(len(data_inp_edit)):
    data_inp_edit[i]=" ".join(data_inp_edit[i])
data_inp_new='¥n'.join(data_inp_edit)+'¥n'
f=open(file_output,'w')
f.write(data_inp_new)
f.close
```

参考文献

1) https://ja.wikipedia.org/wiki/ガラス
2) https://www.asahiglassplaza.net/catalogue/sougou_gijutsu/spdfdata/00041_9s.pdf
3) 山根正之,安井至,和田正道,国分可紀,寺井良平,近藤敬,小川晋永,“ガラス工学ハンドブック普及版”,朝倉書店,1999
4) 黒後貴広、朝光拓也、福山智子、佐藤淳,”ガラス板が拘束された骨組の力学的性状”,日本建築学会学術講演梗概集,2011
5) 本田幾久世、朝光拓也、福山智子、金容善、佐藤淳,”ガラス板が拘束された骨組の設計におけるモデル化の提案”,日本建築学会学術講演梗概集,2012
6) 岩永陽輔、佐藤淳,”ガラス板が拘束された金属骨組の骨格曲線を再現するモデル化”,日本建築学会学術講演梗概集,2013
7) 佐藤淳,”鋼製骨組にガラス板が拘束された構造体の設計法に関する研究”,2013
8) アントニ、佐藤淳,”鋼製骨組にガラス板が拘束された構造体のモデル化における錫緩衝材の剛性と座屈拘束効果”,日本建築学会学術講演梗概集,2014
9) 孫、佐藤淳,”鋼製骨組にガラス板が拘束された構造体のモデル化における座屈拘束効果と Ds 値の設定法”,日本建築学会学術講演梗概集,2015
10) 瀧本信幸,”鋼製骨組にガラス板が拘束された構造体における初期剛性及び座屈荷重の評価法”,東京大学大学院新領域創成科学研究科社会文化環境学専攻 2016 年度修士論文
11) 大霜潤也,”鋼製骨組にガラス板が拘束された構造体における錫緩衝材としての錫の挙動の分析”,東京大学工学部建築学科 2017 年度卒業論文
12) 今井連,”鋼製骨組にガラス板が拘束された構造体におけるガラスの位置変化と解析手法”, 東京大学工学部建築学科 2019 年度卒業論文
13) 西村祐哉,”局所過熱によるガラス溶着法における残留応力の低減法と数値解析アルゴリズム”,東京大学大学院新領域創成科学研究科社会文化環境学専攻 2017 年度修士論文
14) 佐藤淳,”佐藤淳構造設計事務所のアイテム”INAX 出版,2010

謝辞

まず初めに、指導教員である佐藤淳先生にはステンドグラス構造壁という新しいガラスの使われ方を切り拓く研究テーマを与えていただき、また日頃の打合せにおいて的確なご指導を頂き、心より感謝申し上げます。佐藤淳研究室での活動は刺激的かつ挑戦的なものばかりで、その活動の一環に携わらせて頂けたことは何事にも代えがたい経験となりました。

副指導を引き受けていただいた清家剛先生には、研究の進め方や論文の構成についての助言をいただき深く感謝いたします。

また、荒木美香さん、古市渉平さんには日々の打合せや、実験において様々な視点からのアドバイスをいただき、研究を進めていく上で大きな助けとなりました。また、学部時代から度々地下の実験の準備や載荷を手伝っていただき、心から感謝申し上げます。

研究室の先輩、同期、後輩にも大変お世話になりました。先輩である朝原真知子さん、張耕嘉さんは突然の質問にもいつも親身に答えていただきました。同期の小島慎平君、高岡俊一郎君、藤本月穂さん、宮本健太君は皆個性的で、授業や研究室活動を共にできたことは楽しく、心に残る日々でした。そして同じ研究テーマの今井連くんをはじめ研究室の後輩には様々なことを手伝っていただき、特に実験の準備を何度も手伝っていただき本当に有難く、感謝しています。自分にとっても、後輩にとっても忘れることのない実験準備となったことと思います。

そして、最後になりましたが AGC 株式会社の木原幹夫様、株式会社矢嶋の矢澤潤一様をはじめとするステンドグラス構造壁の試験体の材料を提供していただいた皆様のおかげで、実験を行うことができ、研究を進めることができました。心より感謝申し上げます。

2020 年
大霜 潤也