

Development of digital twin of experimental thermal manikin integrated with multi-node thermo-regulation model

馬, 夢臣

九州大学大学院総合理工学府環境エネルギー工学専攻

<https://hdl.handle.net/2324/4785179>

出版情報 : 九州大学, 2021, 修士, 修士
バージョン :
権利関係 :



**Development of Digital Twin of Experimental Thermal manikin
Integrated with Multi-Node Thermo-Regulation Model**

A Thesis

by

Ma Mengchen

Supervisor

Prof. Kazuhide ITO

Master of Engineering

Building Environmental Engineering Laboratory

Department of Energy and Environmental Engineering

Interdisciplinary Graduate School of Engineering Sciences

Kyushu University

Fukuoka·Japan

February 8th, 2022

This page intentionally left blank

ACKNOWLEDGEMENTS

First of all, I would like to thank Prof. Kazuhide Ito for his guidance in these two years. When I first came to Ito's laboratory, I felt the encouragement and trust, which positively impacted my research in the past two years. Thanks to Professor Ito, I never felt discouraged when I encountered problems and could move forward bravely when facing pressure. Coronavirus became quite serious since I first came to Kyushu. Professor Ito was always concerned about my life cared about me, who studied abroad. Professor ITO has been the most potent backing and supporter of my life and research in Kyushu in the past two years. The time in Professor Ito's laboratory will eventually be a beautiful and unforgettable memory in my life.

At the same time, I would also like to thank my co-supervisor, Yoo Sung-Jun. He helped me a lot with my research problems, large and small, from program writing, calculation debugging, article guidance, and concern about my research progress, making me more confident. Without his help and reminder, I could not have had a quick and detailed understanding of each research task, let alone successfully overcome one difficulty after another.

At the same time, I would also like to thank Dr. Kazuki Kuga, Dr. Nguyen Dang KHOA, all the teachers and students in the research room. Because of their vital help and in-depth communication, I can learn from their research experience and complete my research smoothly.

At the same time, I would also like to thank my family for their spiritual and financial support, which enabled me to start my research life at Kyushu University in Japan in the past two years. Because of COVID-19, we haven't seen it for two years, but love from the other side of the ocean has always been able to feel it.

Finally, I would also like to thank my girlfriend, Liu Yuhan. Although we are in different cities, she can always understand me and help me adjust my mood when my research is in trouble. At the same time, I would also like to thank all my friends in my home country for giving me a genuine sense of belonging. Thank you for appearing in my life.

ABSTRACT

This research aims to develop a Computational Simulated Person (CSP) used in a computer simulation. Since the geometric shape and the selected human body thermodynamic model are the same as the Thermal manikin (ANDI)[1] developed by Thermetrics, the CSP model developed this time can also be regarded as the Digital Twin of the Thermal manikin.

Fiala model[2] is used to realize the thermal regulation within tissues of each segment, which describe the heat generation by metabolism, heat transfer between layer, and the heat exchange due to the blood flow. The Fiala thermal model applied to computer calculations is mainly applied to the USER DEFINE FUNCTION of FLUENT[3]. Through its DEFINE_INITIAL and DEFINE_PROFILE macros, the initialization of the calculation parameters of each calculation node of the human body and the iterative calculation of temperature and calculation parameters are realized, and the skin boundary values can be returned to the surface of the geometric model, participating in thermal calculations with the environment side(CFD side). The geometric model of this study is mainly obtained through geometric data acquisition of Thermal manikin through a high-precision handheld scanner, through geometric repair and mesh procession.

By placing Thermal manikin in the chamber and adjusting the ambient temperature, we can obtain the temperature change of the skin surface. After that, we also established a virtual room based on the simulation environment in the computer, placed the CSP model in it at the same position with the same posture, and set the same boundary conditions as the experimental measurement procedure. Experiments are carried out with various variables such as human body posture, indoor air temperature, clothing status to explore whether the CSP model can be used as the Digital Twin of the original Thermal manikin.

Through the validation in several cases, the newly developed CSP model can be used in the environmental thermal simulation calculation by ANSYS FLUENT. The Digital Twin developed in this

study can well reflect the impact of the human body on the environment and the surface temperature of the human body after a series of complex thermal regulations, which is helpful for more accurate analysis for further measurement of human thermal comfort.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS	2
ABSTRACT	3
1 Introduction	7
1.1 Background	8
1.2 Composition of the thesis	10
2 Outline of numerical analysis	11
2.1 Introduction	11
2.2 Numerical analysis of flow field, temperature field, and humidity field	12
2.3 Turbulence model	13
2.3.1 Standard $k-\varepsilon$ model	13
2.3.2 SST $k-\omega$ model	14
2.4 Surface to Surface (radiation model)	16
3 Application of computational thermal theory	18
3.1 Introduction	18
3.2 Thermal model	19
3.2.1 Body construction	19
3.2.2 Heat conduction	20
3.2.3 Metabolism	20
3.2.4 Blood Circulation	21
3.3 Numerical Methods	23
3.3.1 discrete method	23
3.3.2 virtual node	24
3.3.3 core temperature calculation	25
3.3.4 surface temperature calculation	25
3.4 Implementation logic and implementation method	26
4 Geometry model	27

4.1 Introduction.....	27
4.2 3d scan process.....	28
4.3 geometric model building	29
5 Validation of Digital Twin Coupled with Computational Fluid Dynamics.....	33
5.1 introduction	33
5.1.1 In-chamber Measurement	33
5.1.2 CFD software simulation	34
5.2 Simulation result of Digital twin of Thermal manikin.....	35
6 Conclusion	40
REFERENCES.....	41
APPENDICES	42
A.1. The UDF macro code	42

This page intentionally left blank

1 Introduction

1.1 Background

In Japan, all citizens' average indoor living time is 15 hours and 16 minutes, about 63% of a day.[4] Most people's sleep, work, study, entertainment, and daily activities are also carried out indoors. The indoor thermal environment has a far-reaching impact on people's health, comfort, and productivity. The indoor thermal environment affects the feeling of the human body by indoor air temperature, air humidity, air velocity, and radiation heat transfer between the human body and the surrounding environment.

On the one hand, while the environment impacts all aspects of the human body, the human body also affects the environment that cannot be ignored. The human body will continuously carry out metabolism, and the heat, water, and chemicals generated by metabolism will eventually be discharged into the environment. Heat is mainly lost to the environment through convective heat transfer, radiation heat transfer, and latent heat caused by evaporation. While most water is excluded from the body through the urinary system, it will also be discharged into the environment through respiration and sweat glands. The carbon dioxide produced by metabolism will eventually be released into the environment through the respiratory system.

In the artificial environment, many problems related to thermal comfort occur every day. The thermal uncomfortable indoor climate limits the availability and functionality of the space and reduces residents' performance in the workplace. Inadequate air conditioning in the office may cause employees to get sick and reduce the company's productivity. Therefore, the primary purpose of the heating and air conditioning industry is to create hygienic and thermal comfortable spaces for users. Therefore, it is of great significance to fully evaluate the thermal regulation of people to figure out the indoor thermal comfort of the human body under complex conditions.[2]

In order to evaluate the indoor environment and human comfort efficiently and accurately, there are two commonly used methods: Measurement and simulation. Direct measurement of temperature and various thermal indexes of the Thermal manikin in the laboratory is a directive evaluation method that can

directly obtain the required data. However, it is not easy to build an experimental environment that meets the theoretical requirements and accurately controls the variables. The simulation of the indoor thermal environment through the relevant computational fluid dynamics software is a commonly used evaluation method. It has high requirements for the establishment of a geometric model. At the same time, the selection of mathematical models such as fluid calculation and thermodynamic calculation will also affect the accuracy of indoor environment evaluation and human thermal comfort evaluation. However, the flexible control of boundary conditions and the repeatability of simulations is of positive significance to evaluating the indoor environment and human thermal comfort. This study developed a digital twin of an experimental thermal manikin integrated with a multi-node thermo-regulation model based on the above two methods. It evaluated the performance of the digital twin model concerning the measured data of the thermal manikin under the same boundary conditions.

Several multi-segmental, multi-node mathematical models of human thermoregulation have been developed in the past three decades. This study aims to build a multi-segmental, multi-node thermal model which could be compiled to the CFD simulation. Fiala model[2] carries out a more detailed division of human body parts and human tissue than the 2-node model[5] and the Stolowijk model[6]. Some calculation method of the Fiala thermal model is adopted in the calculation method. It can more accurately reflect the thermal regulation inside the human body. This research aims to integrate its mathematical model into a computer-simulated person (CSP) compiled to the CFD simulation. In this study, Ansys Fluent[7] is mainly used for thermal simulation of the environment. Through UDF[3] programming, the boundary conditions of the human body can be flexibly controlled. The UDF code calculates various thermal regulatory activities inside the human body, and the temperature value is calculated at each iteration. To each computing node, the UDF function finally returns the skin temperature and humidity of the human body to the surface of the geometric human body model as boundary conditions. The heat movement (convection, radiation, latent heat) generated from the surface of the human body is calculated by the software (Ansys Fluent) side. The coding of human thermal models based on numerical models such as Fiala is the core of our research.

After establishing the CSP model based on the Fiala model, we still need the geometric model as a carrier to complete its simulation calculation in the CFD software. The geometry of Thermal manikin developed by the American company Thermetrics (www.thermetrics.com) was selected for this study, which internal thermal regulation is also based on Fiala's mathematical model. The thermal model developed in this study was combined with the geometric model and compared with the experimental measurement results of Thermal manikin in the same environment in the laboratory. Suppose the accuracy of the Digital Twin can be verified. In that case, it can be used for simulation under various experimental conditions that require the participation of the human thermal model, and the advantages of CFD simulation calculation compared with experimental measurement can be exerted.

To validate the Digital Twin, body surface temperature, heat flow, and other parameters of the original human thermal model under a specific environment with the corresponding parameters of the computer simulation results based on the same environment will be compared to the experimental measurement where the Thermal manikin is located.

1.2 Composition of the thesis

The article mainly includes four parts. The first part is the outline of numerical analysis, which is introduced in chapter 2. The second part is shown in chapter 3, which mainly talks about the application of computational thermal theory. The third part is about establishing the geometry model in chapter 4. Furthermore, the last part will discuss the validation of Digital Twin (CSP) in chapter 5.

The mathematical models and formulas used in CFD calculation involved in the research will be included in the second chapter. It includes the mathematical models involved in the numerical analysis of flow field and temperature field, the mathematical models involved in turbulence calculation, and the mathematical models involved in calculating direct heat radiation at each boundary.

The establishment of the Digital Twin (CSP) model is mainly divided into the application of the human body thermal model and the establishment of the human geometric model. The third chapter mainly introduces the human thermal model involved in developing the Digital Twin (CSP) model. This paper refers to the mathematical thermal model used in the Fiala model, arranges its governing equations, and writes them as UDF code that can be used by CFD software (ANSYS fluent). This chapter mainly introduces the generation and movement of heat inside the human body and the determination of the human surface's thermal and wet boundary conditions. According to the laboratory environment, the room model and manikin in the geometric model are established. The room is modeled with the same size as the chamber where the measurement was carried out. A 3D scanner was used to obtain the CAD data of the Thermal manikin. The Digital Twin (CSP) model reflecting the thermal regulation inside the human body refers to the Fiala model, the same as the Thermal manikin.

The fifth chapter mainly introduces the validation of the Digital Twin model. The Thermal manikin model is placed in the laboratory chamber, and the air conditioning system controls the indoor air temperature. The simulation environment of the CSP model adopts the same geometric model as the measured room, and the position of the CSP model is placed in the same position as the measured room as far as possible. The skin temperature changes and distribution of various parts of the Thermal manikin were detected in the measured environment, and the temperature data of various parts of the human body were collected hourly in the simulated environment. By comparing the measured and simulated results, the performance of the CSP model can be evaluated.

2 Outline of numerical analysis

2.1 Introduction

This study needs to verify whether the newly designed CSP model can realize the functions of the original model(Fiala's model) and discuss its performance in simulation calculation. We need to compare the thermal manikin's flow field and temperature distribution in the actual experimental environment

with the Simulation result based on the CSP model, which could be seen as a digital twin of the thermal manikin in the simulated environment. Therefore, the high-precision simulation of CFD plays a vital role in this study. Accurate numerical analysis is required. Thus, ANSYS FLUENT[7], a commercial CFD analysis software, was used in this research and based on four aspects. (1) flow field analysis for indoor airflow velocity distribution and convection heat transfer analysis. (2) Air temperature distribution based on the flow field and human body surface temperature distribution. (3) coupled analysis of radiant heat transfer by S2S (Surface to Surface) radiation model.

2.2 Numerical analysis of flow field, temperature field, and humidity field

The fluid density in the experiment adopts Boussinesq approximation and considers air incompressible fluid(2-1). This method assumes that the change of density affects the flow field, and only the buoyancy needs to be considered in the calculation. This method has a low computational cost and easy convergence. When the density change is not apparent because the nonlinearity of the problem is reduced, accurate results can also be obtained by using this approximation.

$$\frac{\partial U_i}{\partial x_i} = 0 \quad (2-1)$$

Navier-Stokes equation(2-2) is used to describe fluid motion. Where u is the fluid velocity, P is the fluid pressure, ρ is the fluid density, and ν is the fluid kinematic viscosity. The Navier-Stokes equation can be considered Newton's second law for fluid motion, representing momentum conservation, while the continuity equation represents the conservation of mass. The two equations need to be solved jointly.

$$\frac{\partial U_i}{\partial t} + U_j \cdot \frac{\partial U_i}{\partial x_j} = -\frac{1}{\rho} \cdot \frac{\partial P}{\partial x_i} + \nu \frac{\partial}{\partial x_j} \left(\frac{\partial U_i}{\partial x_j} + \frac{\partial U_j}{\partial x_i} \right) - g_i \beta \theta \quad (2-2)$$

In a non-isothermal flow field, the energy transfer in the temperature field can be described according to the following equation(2-3). α represents the thermal diffusion coefficient, and S is the heat source term. This equation follows Fourier's law and expresses the conservation of energy.

$$\frac{\partial \theta}{\partial t} + \frac{\partial U_j \theta}{\partial x_j} = \frac{\partial}{\partial x_j} \alpha \frac{\partial \theta}{\partial x_j} + S \quad (2-3)$$

2.3 Turbulence model

The turbulence models widely used in the RANS approach are two-equation models such as k - ω and k - ε models. In the two-equation model, Reynolds stress and turbulent heat flux are expressed by the Boussinesq hypothesis as follows

$$-\overline{u_i u_j} = 2\nu_t S_{ij} - \frac{2}{3} k \delta_{ij} \quad (2-4)$$

$$-\overline{\theta u_j} = \frac{\nu_t}{\sigma_\theta} \frac{\partial \theta}{\partial x_j} \quad (2-5)$$

σ_θ is the turbulent Prandtl number for energy. In equations (2-4) and (2-5), the kinematic viscosity and temperature diffusivity are locally increased by the eddies. The buoyancy effect is omitted in the following description.

2.3.1 Standard k - ε model

The transport equation of the turbulent kinetic energy k and its dissipation rate ε is expressed as follows.

$$\frac{\partial k}{\partial t} + \frac{\partial (k U_i)}{\partial x_i} = \frac{\partial}{\partial x_j} \left[\left(\nu + \frac{\nu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right] + G_k - \varepsilon \quad (2-6)$$

$$\frac{\partial \varepsilon}{\partial t} + \frac{\partial (\varepsilon U_i)}{\partial x_i} = \frac{\partial}{\partial x_i} \left[\left(\nu + \frac{\nu_t}{\sigma_\varepsilon} \right) \frac{\partial \varepsilon}{\partial x_i} \right] + C_{1\varepsilon} \frac{\varepsilon}{k} G_k - C_{2\varepsilon} \frac{\varepsilon^2}{k} \quad (2-7)$$

The eddy viscosity coefficient ν_t is calculated from k and ε as follows.

$$\nu_t = C_\mu \frac{k^2}{\varepsilon} \quad (2-8)$$

Each model constant has the following values are:

$$C_{1\varepsilon} = 1.44, C_{2\varepsilon} = 1.92, C_\mu = 0.09, \sigma_k = 1.0, \sigma_\varepsilon = 1.3 \quad (2-9)$$

G_k term is:

$$G_k = -\overline{u_i u_j} \frac{\partial u_j}{\partial x_i} \quad (2-10)$$

$$G_k = \nu_t S^2 \quad (2-11)$$

$$S \equiv \sqrt{2S_{ij}S_{ij}} \quad (2-12)$$

2.3.2 SST k - ω model

SST is a modified model in which the equation for shear stress transport is added to the Baseline k - ω model modified to solve the free flow with the above-mentioned k - ω model and the standard k - ω model proposed by Wilcox at the boundary layer. It is called the k - ω model. The k - ω model is an equation related to the turbulent energy k and the specific dissipation rate $\omega = \varepsilon / k$, and each transport equation is expressed by the following equation.

$$\frac{\partial k}{\partial t} + \frac{\partial(kU_i)}{\partial x_i} = \frac{\partial}{\partial x_j} \left[\left(\nu + \frac{\nu_t}{\sigma_k} \right) \frac{\partial k}{\partial x_j} \right] + G_k - Y_k \quad (2-13)$$

$$\frac{\partial \omega}{\partial t} + \frac{\partial(\omega U_i)}{\partial x_i} = \frac{\partial}{\partial x_j} \left[\left(\nu + \frac{\nu_t}{\sigma_\omega} \right) \frac{\partial \omega}{\partial x_j} \right] + G_\omega - Y_\omega \quad (2-14)$$

The vortex viscosity coefficient ν_t is calculated from k and ω as follows.

$$\nu_t = \alpha^* \frac{k}{\omega} \quad (2-15)$$

G_k is defined by the equation(2-10) as in the k - ε model, but a limiter is provided by the following equation to suppress the excessive generation of turbulent energy near the stagnation point.

$$G_k = \min(G_k, 10\beta^* k \omega) \quad (2-16)$$

Since the equation limits the vortex viscosity coefficient(2-28) in the SST k - ω model, the generation term G_ω of ω given by the following equation is important.

$$G_\omega = \frac{\alpha \alpha^*}{\nu_t} G_k \quad (2-17)$$

Also, the dissipation of k is:

$$Y_k = \beta^* k \omega \quad (2-18)$$

Dissipation of ω is:

$$Y_\omega = \beta_i \omega^2 \quad (2-19)$$

In the SST k - ω model, the following equation with the new cross-diffusion term D_ω added to the right side of the equation (2-14) is solved as the transport equation of ω .

$$\frac{\partial \omega}{\partial t} + \frac{\partial(\omega U_i)}{\partial x_i} = \frac{\partial}{\partial x_j} \left[\left(\nu + \frac{\nu_t}{\sigma_\omega} \right) \frac{\partial \omega}{\partial x_j} \right] + G_\omega - Y_\omega + D_\omega \quad (2-20)$$

$$D_\omega = 2(1 - F_1) \frac{\sigma_{\omega,2}}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j} \quad (2-21)$$

It is switched to solve the k - ε model when $F_1 = 0$ and the k - ω model when $F_1 = 1$.

$$\sigma_k = \frac{1}{F_1 / \sigma_{k,1} + (1 - F_1) / \sigma_{k,2}} \quad (2-22)$$

$$\sigma_\omega = \frac{1}{F_1 / \sigma_{\omega,1} + (1 - F_1) / \sigma_{\omega,2}} \quad (2-23)$$

$$\beta_i = F_1 \beta_{i,1} + (1 - F_1) \beta_{i,2} \quad (2-24)$$

$$F_1 = \tanh(\Phi_1^4) \quad (2-25)$$

$$\Phi_1 = \min \left[\max \left(\frac{\sqrt{k}}{0.09 \omega y}, \frac{500 \nu}{\omega y^2} \right), \frac{4 \rho \sigma_{\omega,2} k}{D_\omega^+ y^2} \right] \quad (2-26)$$

$$D_\omega^+ = \max \left[2 \frac{\sigma_{\omega,2}}{\omega} \frac{\partial k}{\partial x_j} \frac{\partial \omega}{\partial x_j}, 10^{-10} \right] \quad (2-27)$$

The limiter is applied to the formula of the vortex viscosity coefficient as follows:

$$\nu_t = \frac{a_1 k}{\max[a_1 \omega, SF_2]} \quad (2-28)$$

$$F_2 = \tanh(\Phi_2^2) \quad (2-29)$$

$$\Phi_2 = \max \left[\frac{2\sqrt{k}}{0.09\omega y}, \frac{500\nu}{\omega y^2} \right] \quad (2-30)$$

Each model constant has the following values:

$$\alpha^* = 1.0, \alpha = 0.52, \beta^* = 0.09, a_1 = 0.31$$

$$\sigma_{k,1} = 1.176, \sigma_{k,2} = 1.0, \sigma_{\omega,1} = 2.0, \sigma_{\omega,2} = 1.168, \beta_{i,1} = 0.075, \beta_{i,2} = 0.0828$$

2.4 Surface to Surface (radiation model)

Radiation calculation is performed by the Surface-to-surface (S2S) model implemented in the commercial CFD code ANSYS Fluent used in this study.

In the S2S radiation model, radiation exchange is calculated with each surface in the enclosed space as a gray diffusion surface, and the absorption, radiation, and scattering of radiation in the space can be ignored, and the energy exchange between the two surfaces is the size of the surface. It depends on the distance between the surfaces and the direction of the surface. These parameters are determined by the view factor calculated from the geometrical positional relationship of each surface.

The emissivity and absorptivity of the gray surface are independent of the wavelength, and the emissivity ε and the absorptivity α are equal according to Kirchhoff's law. Also, the reflectance ρ on the diffusion surface is independent of the reflection or incident direction.

In the gray diffusion model, the radiant energy exchange between the surfaces to be analyzed is not affected by the medium that separates the surfaces, so when a certain amount of radiant energy E is incident on the surface, it is reflected at a specific rate (ρE), Absorbed (αE), and transmitted (τE). Since the surface to be analyzed has opacity, the transmittance can be ignored. According to the law of conservation of energy, $\rho = 1 - \varepsilon$ from the above relationship ($\varepsilon = \alpha$).

The energy flux generated from a surface consists of radiant energy and reflected energy. Since the reflected energy flux depends on the incident energy flux, the energy flux generated from the surface k is expressed as follows:

$$q_{out,k} = \varepsilon_k \sigma T_k^4 + \rho_k q_{in,k} \quad (2-31)$$

Where $q_{out,k}$ is the energy flux leaving the surface, ε_k is the emissivity, σ is the Stefan-Boltzmann constant, and $q_{in,k}$ is the energy flux incident on the character from the surroundings.

The amount of incident energy upon a surface from another surface directly affects the surface to surface view factor. The view factor is the fraction of energy leaving the surface incident on a surface. The characters used in calculating a view factor can be mesh faces or (for 3D cases only) clusters of faces; see Clustering and View Factors for details about groups. The incident energy flux can be expressed in terms of the energy flux leaving all other surfaces as:

$$A_k q_{in,k} = \sum_{j=1}^N A_j q_{out,j} F_{jk} \quad (2-32)$$

There is a view factor between surface and surface. For characters, using the view factor reciprocity relationship gives:

$$A_j F_{jk} = A_k F_{kj} \quad (2-33)$$

So that:

$$q_{in,k} = \sum_{j=1}^N F_{jk} q_{out,j} \quad (2-34)$$

Therefore:

$$q_{out,k} = \varepsilon_k \sigma T_k^4 + \rho_k \sum_{j=1}^N F_{jk} q_{out,j} \quad (2-35)$$

3 Application of computational thermal theory

3.1 Introduction

In the research of indoor human thermal comfort evaluation or indoor pollutant diffusion analysis, the simulation calculation based on CFD has apparent advantages. A multi-segment, multi-node Computer Simulated Person (CSP) model was developed. The detailed division of various parts of the human body and the radial division of the same part can more realistically reflect the thermal regulation of the human body. The human body thermal model was used in this study mainly refers to the Fiala Model because its theoretical framework and calculation methods have many merits compared with the previous 2-Node[5] model developed by Gagge and Stolowijk model[6], the Fiala model also has more accurate theoretical prediction accuracy.

Heat will be generated inside the human body through metabolism, and heat will be transferred through heat conduction between various body tissues. On the one hand, heat will be lost to the environment on the skin surface, mainly including convection, radiation, water evaporation, etc. On the other hand, heat will also be transferred to various body parts through blood flow. With the saving and dissipation of heat, the temperature of human tissues and even the human body's core will increase or decrease.

The human thermal model can govern the heat generation and heat transfer inside the human body in the CFD simulation. At each iteration of the calculation, the human body's thermal and humidity boundary conditions are returned to the geometric model of the human body for joint analysis with the environmental side. Therefore, a thermal model that can truly reflect the natural human body is significant for indoor CFD simulation.

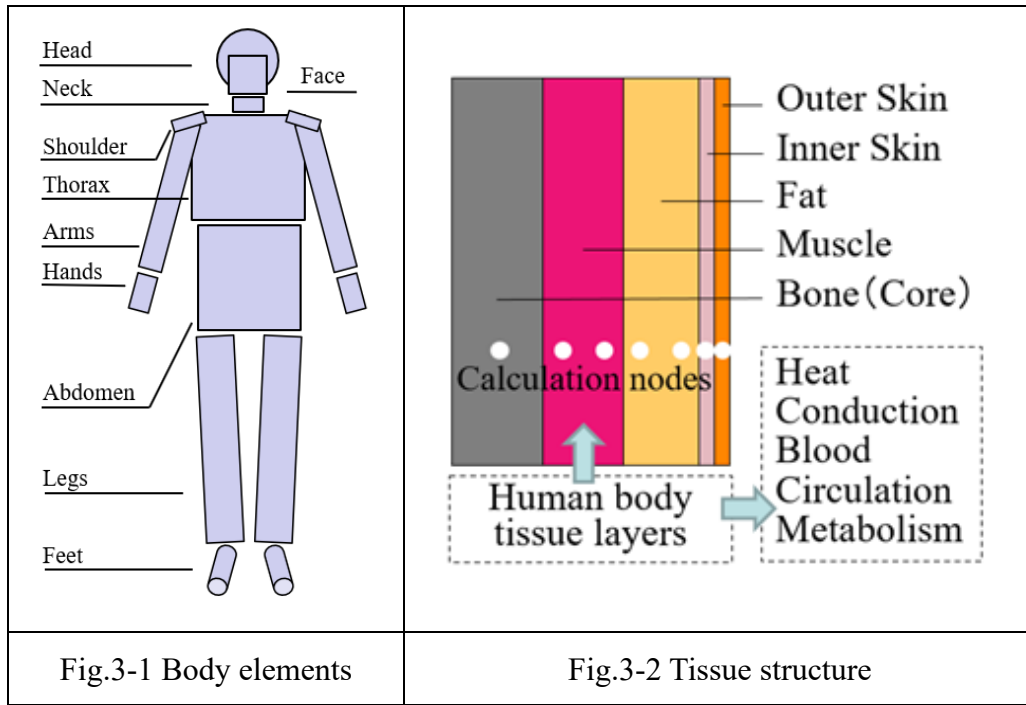
This research aims better to apply the human thermal model to simulation calculations. The Fiala model, the same as the thermal manikin, was selected as the human body thermal model. On the one hand, the Fiala model is significantly optimized than the original Stolowijk and other human thermal models. On

the other hand, the digital twin based on the same thermal model of the human body can be compared to the measurement result to reach validation.

3.2 Thermal model

3.2.1 Body construction

The body model selected in this study is a man model with a bodyweight of 73.5 kg, a body surface area of 1.9 square meters, and a basal metabolic rate of 87W. The human body is divided into 15 parts, which are cylindrical or spherical in shape, including head, face, neck, shoulders, arms, hands, thorax, abdomen, legs and feet, shown as Fig3-1.



Pennes described the mechanism of heat transfer in living tissue in the so-called bio thermal equation[8].

This differential equation describes heat dissipation in a homogeneous, infinite tissue volume:

$$k \left(\frac{\partial^2 T}{\partial r^2} + \frac{\omega}{r} \frac{\partial T}{\partial r} \right) + q_m + \rho_{bl} w_{bl} c_{bl} (T_{bla} - T) = \rho c \frac{\partial T}{\partial t} \quad (3-1)$$

The Fiala model is a multi-node thermal model. The number of nodes can be defined in different tissue layers in the radial direction to meet calculation accuracy and speed requirements. Different computing nodes obey Pennes' thermal equation, which considers heat Conduction, blood circulation and

metabolism.

3.2.2 Heat conduction

The heat conduction term in the equation (3-1) considers only temperature variations in the radial direction, whereby $\omega=1$ stands for polar coordinates and $\omega=2$ for spherical coordinates (head). The computational unit in this study is an irregular fan-like geometry delimited by the facets of the mannequin skin surface to the human core. Since the main direction of heat conduction is from the core to the environment, the heat conduction between the sectors can be ignored, so the heat conduction in this study only considers the radial heat conduction. References Fiala papers and references in the equation.

3.2.3 Metabolism

The model represents the total metabolic heat production of different computing nodes in different parts by q_m , mainly two parts. The first part is basal metabolism, mainly to maintain the normal physiological state of cells. The second part is additional heat production, represented by Δq_m , mainly includes exercise heat production, shivering heat production, *etc.*

$$q_m = q_{m,bas,0} + \Delta q_m \quad (3-2)$$

Fiala's Model can obtain the basal metabolic rate $q_{m,bas,0}$ [2]. At thermal neutrality, $M_{bas,0} = \int q_{m,bas,0}$ The total metabolism of the human body is 87W, which meets the metabolic rate of the human model in ASHRAE Standards 55 and ISO 7730.

$$\Delta q_m = \Delta q_{m,bas} + \Delta q_{m,sh} + \Delta q_{m,w} \quad (3-3)$$

$\Delta q_{m,bas}$ expresses the actual basal metabolism and the difference between the basal metabolism in the neutral thermal conditions. If the tissue temperature is cha with the temperature setpoint T_0 , $q_{m,bas}$ is also present in the non-muscle tissue.

The $q_{m,w}$ term can be written as:

$$q_{m,w} = \frac{\partial(a_{m,w}H)}{\partial V_{msc}} \quad (3-4)$$

Where $a_{m,w}$ are the coefficients that allocate the overall additional metabolism due to work, V_{msc} is the corresponding body element muscle volume, and H is the internal whole-body workload. The $a_{m,w}$ values for standing activities are from Stolwijk[6]. This model uses other estimates of $a_{m,w}$ for sedentary activities because additional metabolism is transferred from the leg muscles to the trunk and arm musculature during sitting. The amount of work H in the equation. Equation(3-4) Represents the fraction of the total additional energy produced by muscle activity but not manifested as external work. It is obtained as the difference between the actual total heat production rate remaining in the body and the basal metabolic $M_{bas,0}$:

$$H = act \frac{m_{bas,0}}{act_{bas}} (1-\eta) - M_{bas,0} \quad (3-5)$$

The calculation of the actual metabolic rate utilizes the activity level units act (measured in met) as related to its basal value of $act_{bas}=0.8$ met.

The bracked term $(1-\eta)$ in the equation (3-5) indicates the fraction of the total metabolic heat produced to heat the body. Not converted into external mechanical power. Human mechanical efficiency is not constant in reality but rises with activity levels. In this model, the following formulas are derived from regression analysis on the measurements of Wyndham et al. [9].

$$\eta = 0.2 \cdot \tanh\{b_1 \cdot act + b_0\} \quad (3-6)$$

3.2.4 Blood Circulation

Blood circulation is significant for heat dissipation in the human body. If there is no blood circulation, the body's temperature of various structures and tissues will rise to an excessive level. In the current model, the human blood circulation system is simulated as three main components: 1) the central blood pool, 2) counter-current heat exchanges and 3) pathways to individual tissue nodes. All body parts are supplied with warm blood from the central pool by the main arteries. Before the tissues of the limbs and shoulders are perfused, the blood is cooled by the counter-current flow. Heat will lose into adjacent veins.

According to the biothermal equation(3-1), arterial blood exchanges heat by convection in the capillary bed. The venous blood then accumulates in the central vein and is reheated by the heat of the adjacent artery when it returns to the central blood pool. It mixes with blood from other elements to produce a new central blood pool temperature.

Counter-current heat exchange (CCX) simulation is introduced into the model to obtain a more realistic arterial blood temperature T_{bla} distribution, rather than assuming that all body components' constant arterial blood temperature is equal to the temperature of the central blood pool T_{blp} . There are many CCX models between adjacent vessel pairs in the literature [10], but these models are rarely included in the systemic model.

Assuming mass-continuity in blood vessels, the decrease in the temperature of the blood in the arteries of an element $T_{blp} - T_{bla}$ corresponds to the increase in temperature $T_{blv,X} - T_{blv}$ of the blood in the veins of the same body element after CCX :

$$\rho_{bl}c_{bl} \int w_{bl} dV (T_{blp} - T_{bla}) = \rho_{bl}c_{bl} \int w_{bl} dV (T_{blv,X} - T_{blv}) \quad (3-7)$$

Here, $\int w_{bl} dV$ represents the blood flow in the veins as the volume-integral of capillary blood flows w_{bl} over the body element. The blood temperatures after undergoing CCX, T_{bla} and $T_{blv,X}$ are the two unknowns in eq.(3-7). In order to obtain an element's arterial blood temperature T_{bla} in eq. (3-7). The net heat exchange between adjacent vessels Q_x was defined using the equation of Gordon[11]:

$$Q_x = h_x (T_{bla} - T_{blv}) \quad (3-8)$$

H_x is the corresponding counter-current heat exchange coefficient and T_{blv} is the venous blood temperature before CCX. The arterial blood temperature T_{bla} of a body element is then obtained by substituting the right-hand side of eq. (3-7) by the right-hand side of eq. (3-8) and rearranging:

$$T_{bla} = \frac{\rho_{bl}c_{bl} \int w_{bl} dV \cdot T_{blp} + h_x \cdot T_{blv}}{\rho_{bl}c_{bl} \int w_{bl} dV + h_x} \quad (3-9)$$

The computation of the central blood pool temperature T_{blp} utilized a special calculation process. Since the bioheat equation assumes that capillary blood reaches an equilibrium with the surrounding tissue, T_{blv} of a body element in eq. (3-9) is obtained as follows:

$$T_{blv} = \frac{\int w_{bl} T dV}{\int w_{bl} dV} \quad (3-10)$$

The heat-exchange coefficients h_x between adjacent arteries and veins in eq.(3-9) are zero for central body elements[11]. Few reliable data on h_x of extremities exist in the literature. Therefore, h_x of extremities and shoulders were estimated by a trial and error procedure in which the coefficients were adjusted to obtain agreement between local skin temperatures as predicted and measured. For legs, feet, arms and hands h_x were concluded by Fiala from many experiments (see Tab.A.1 of [2]). In non-neutral conditions or when working, the blood flows, w_{bl} in eq.(3-1), vary with changes in regional metabolic rates, q_m . Experimental results [12] indicate a linear relationship between w_{bl} and q_m . Instead of dealing with blood perfusion rates w_{bl} , the present model considers directly their energy equivalent $\Delta\beta = \rho_{bl}c_{bl}w_{bl}$ [$W \cdot m^{-3}K^{-1}$]. The increased demand for oxygen is therefore accounted for by calculating the change in the gain factor $\Delta\beta = \rho_{bl}c_{bl}w_{bl}$ as a function of the change in metabolism q_m using a proportionality constant $\mu_{bl} = 0.932 [K^{-1}]$ which was obtained from Stolwijk[6]:

$$\Delta\beta = \mu_{bl} \cdot \Delta q_m \quad (3-11)$$

Utilizing the blood perfusion term in eq. (3-1) becomes:

$$\beta(T_{bla} - T) \text{ where } \beta = \beta_0 + \Delta\beta \text{ and } \beta_0 = \rho_{bl}c_{bl}w_{bl,0} \quad (3-12)$$

3.3 Numerical Methods

3.3.1 discrete method

In this model, the computational unit is a facet. Each facet in the radial direction will be assigned several

computing nodes. The temperature close to the outside is usually the steepest and most prone to error in temperature calculation. Therefore, more computing nodes will be allocated to the outer layer during node allocation to maintain the calculation accuracy.

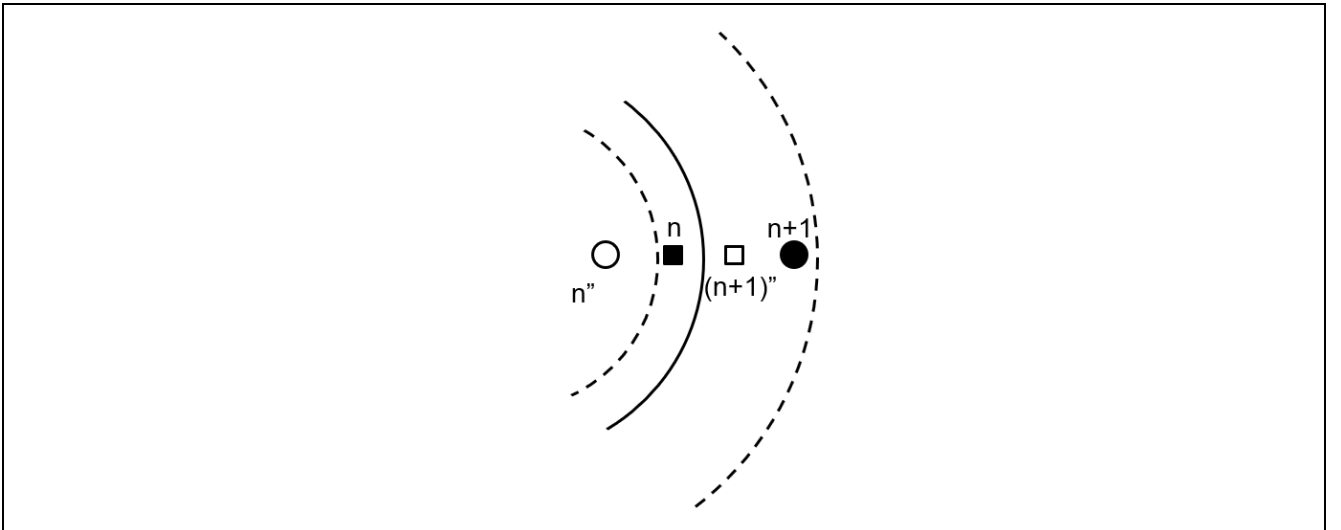
The calculation of nodes is discretized by finite difference. Compared with the first-order accuracy of forward or backward difference, the central difference method can obtain the second-order accuracy and more accurate results. The calculation adopts explicit calculation, and the node's parameters (temperature) of the next time are obtained through the parameter conditions of the current time.

$$\rho_r c_r \frac{T_r^{(t+1)} - T_r^{(t)}}{\Delta t} = k_r \left(\frac{T_{r+1}^{(t)} + T_{r-1}^{(t)} - 2T_r^{(t)}}{\Delta r^2} + \frac{T_{r+1}^{(t)} - T_{r-1}^{(t)}}{2r\Delta r} \right) + q_{m,r}^{(t)} + \beta_r^{(t)} (T_{bla}^{(t)} - T_r^{(t)}) \quad (3-13)$$

3.3.2 virtual node

Considering the interface between the n th layer and the $n+1$ th layer, the last computing node of the upper layer has physical parameters such as r_n , d_{rn} , k_n , and the first computing node of the next layer has r_{n+1} , d_{rn+1} , k_{n+1} and other physical parameters. Considering the conservation of energy flowing through the interface, we have the following equation:

$$\Delta\varphi \frac{k_n}{\ln\left(\frac{r_{(n+1)'}}{r_n}\right)} (T_{r_n} - T_{r_{(n+1)'}}) = \Delta\varphi \frac{k_{n+1}}{\ln\left(\frac{r_{n+1}}{r_{n''}}\right)} (T_{r_{n''}} - T_{r_{n+1}}) \quad (3-14)$$



3.3.3 core temperature calculation

Since this calculation only considers the heat transfer in the radial direction, the temperature gradient calculated from the facets of the body surface to the core is not symmetrical. The axial (center) temperature of each part as a boundary condition is of great significance for calculating the temperature of each node, so the setting of the axial temperature boundary condition also needs to be considered. When calculating the heat transfer term of the next node, the radius of the previous node will be used as the denominator in the calculation. Therefore, the core axis cannot ignore the radius. In this study, the radius of the core axis is set to be 1/10 of the core tissue.

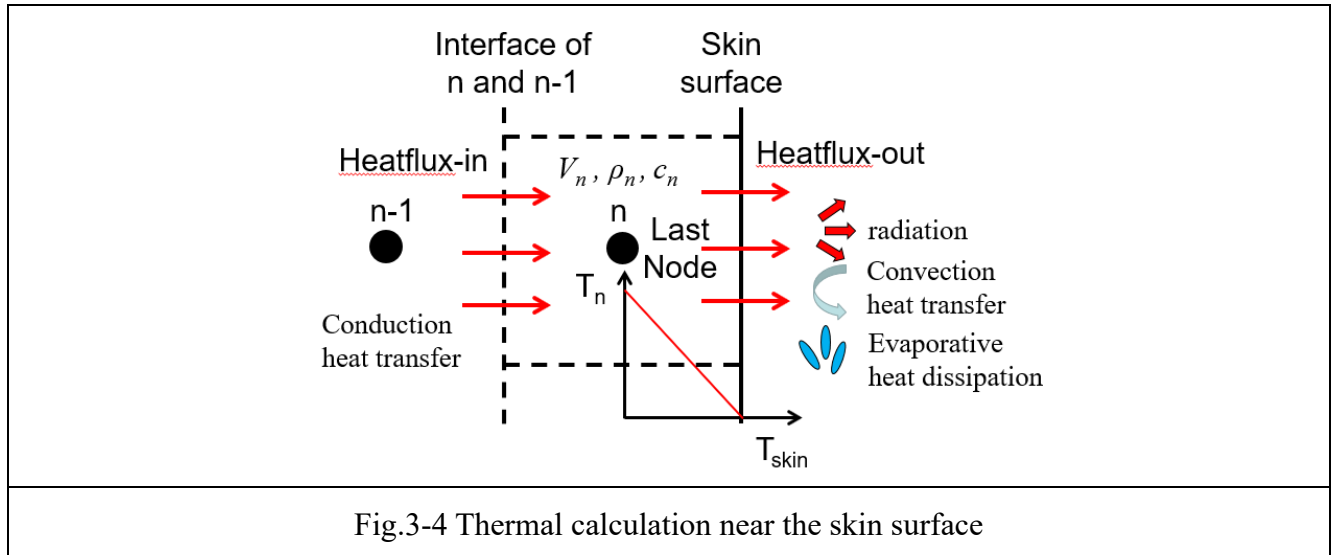
3.3.4 surface temperature calculation

The temperature calculation of each node in the tissue layer is mainly determined by the temperature of the front and rear nodes. However, at the last calculation node of each part of the body, the next node is in the air, and we cannot continue to perform calculations by creating virtual nodes. This study's calculation is based on the control volume concept where the last calculation node is located through energy conservation theory.

Since the control volume where the last calculation node is located belongs to the outer skin, there is no blood flow and metabolism in this tissue, so we only need to consider the thermal conduction term. Since this study regards the internal heat conduction of the human body as one-dimension heat conduction, we only need to consider the radial heat transfer. For each control volume, we first need to obtain the heatflux-in and heatflux-out at both ends, and the interpolation is used as the net increase of its net heat. Heatflux-in is the heat flow of the interface between n nodes and n-1 nodes, and heatflux-out is obtained through BOUNDARY_HEAT_FLUX FUNCTION in UDF.

After obtaining the net heat gain of the control volume, calculate the temperature of the last node by

formula(3-1). The temperature gradient from the skin to the last node is considered a linear change, and Fourier's law of heat conduction finally calculates the skin temperature. The calculated skin temperature is used as a boundary condition to participate in the heat exchange calculation with the environment. Fig.3-4 shows the process calculation of the temperature of the last node and the skin surface temperature.



3.4 Implementation logic and implementation method

Integrating the CSP model with CFD calculation to validate the performance of Thermal manikin is an essential subject of this research. Based on ANSYS FLUENT, this research adopts Fiala's multi-node human body thermal model with the surface of the geometric model as boundary conditions to participate in numerical calculations.

In this research, we use a user-defined function (UDF, User Define Function) that allows the user to customize boundary conditions, physical property values, etc., and incorporate it into the introduction of the multi-node model. Figure.3-4 shows the application and calculation process of the multi-node model using UDF.

To explain each UDF macro used to introduce the multi-Node model, the DEFINE_INIT macro is a macro that defines the initial conditions for analysis. Enter the initial parameters required to calculate the multi-Node model in the cell. Therefore, this DEFINE_INIT macro works only once at the beginning.

On the other hand, the DEFINE_PROFILE macro is a macro that is repeatedly applied for each analysis iteration. For example, reading the node temperature and skin surface temperature in the previous step and data from memory. Furthermore, a macro defines the boundary conditions for analysis, and the skin surface temperature calculated from the multi-Node model is set using DEFINE_PROFILE.

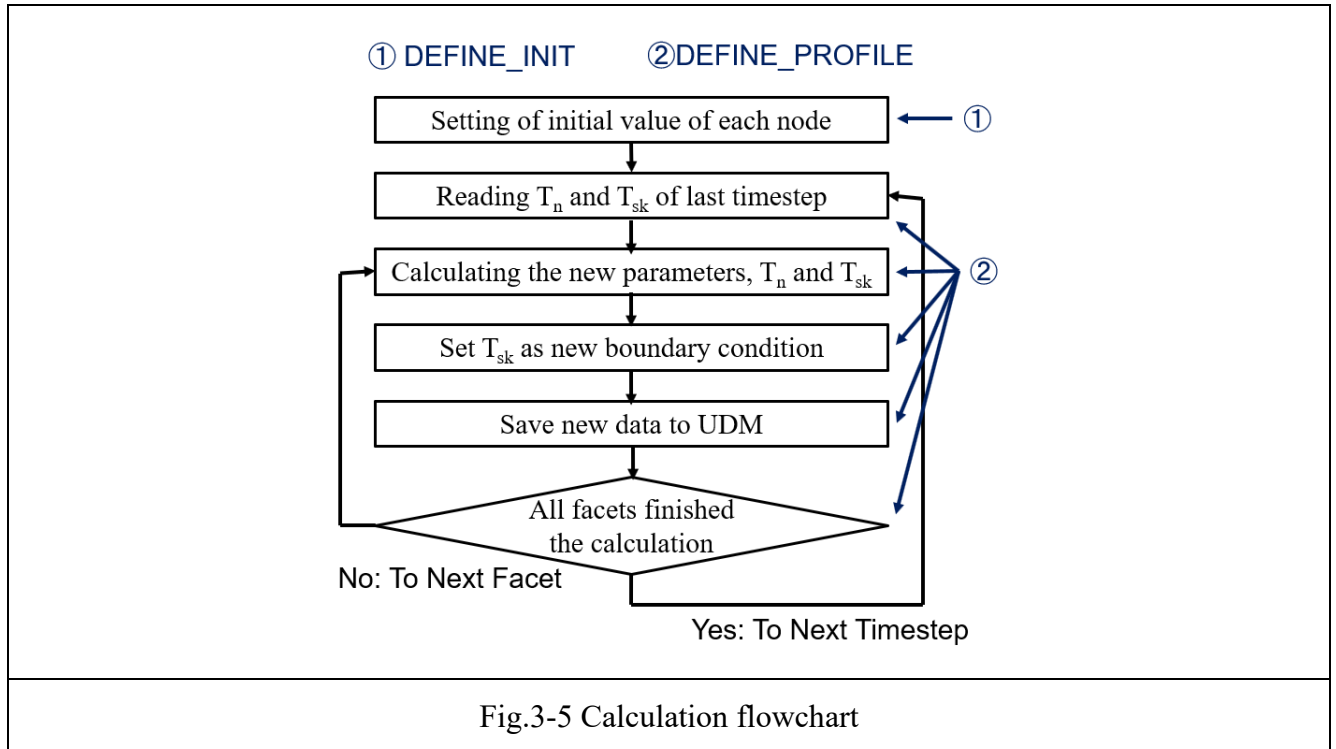


Fig.3-5 Calculation flowchart

4 Geometry model

4.1 Introduction

The geometric model is also fundamental in CFD calculation. A computational geometry model is an idealized digital representation of an object's shape. An excellent geometric model needs to represent the main geometrical character of the simulation object as much as possible so that the simulation calculation is closer to the actual situation. This study aims to verify the digital twin model established based on the Fiala model and compare it with the thermal manikin in the laboratory. In order to control the variables, the simulated and measured experiments adopt the same environmental boundary conditions and the same geometric information of the human body. Therefore, we scanned the 3-D geometry data of thermal manikins and used it in the geometric modeling of the CSP model.

4.2 3d scan process

In order to obtain high-precision geometric data of the human body model, we chose a 3D scanner (EinScan Pro 2X Plus) from SHINING 3D. It Offers an enhanced Handheld HD Scan Mode and enlarged scan range and could capture 3D models of medium to large size objects with high efficiency.

The manikin to be scanned in this study has complex geometry and includes rich small structures, so it is not easy to obtain all geometric data through the fixed scan. Therefore, the high-precision mode of handheld HD scan is adopted in this study. The error can be controlled within $0.05\text{mm} \pm 0.3\text{mm/m}$.



Fig.4-1 3D Scanner (EinScan Pro 2X Plus by SHINING 3D)

The geometric model is a multi-segment human body thermal model developed by Thermetrics Company in the United States. The human thermal model divides the human body into multiple parts: head, limbs, chest, abdomen, face, and neck. It refers to the Fiala human body thermal model, and further division of parts is carried out on its basis. Since the CSP model developed based on Fiala refers to the original part division in Fiala's paper[2], the digital geometric model will be based on Thermal Manikin's geometric outline and Fiala Model's part division.



Fig.4-2 Naked thermal manikin



Fig.4-3 clothes thermal manikin

4.3 geometric model building

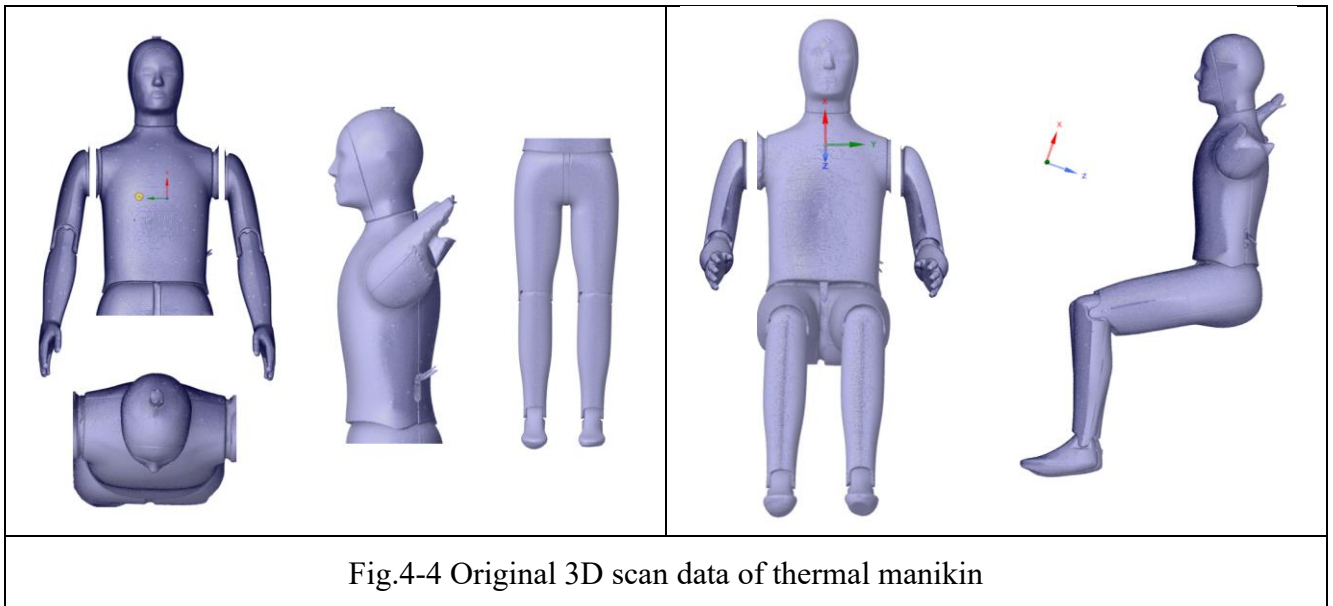
4.3.1 Naked model

Firstly, the naked human model uses a 3D scanner to scan the whole model with high precision and obtain its CAD data. Since the model is in the suspended state, the thermal manikin is prone to displacement after long-time suspension. Moreover, the internal measurement of the arm has a large angle limit on 3D laser scanning. Compared with one-time scanning of the whole human body, this study separately scanned the manikin's arms, upper body, and lower body. In the post-processing stage of the model, all parts will be spliced.

Read the CAD data of the model through space claim software. In order to apply the model to more scenes and simulate more human postures, the joints of the original thermal manikin were designed as movable hinges. In establishing a digital geometric model, the details of the joint hinge do not belong to the details of the real human body. At the same time, it will also cause a great burden on mesh generation.

After reading the scanning data, the whole body joints are smoothed.

To ensure the consistency of part segmentation between the experimental and analog calculation models, the digital geometric model refers to the geometric segmentation line of the thermal manikin and uses the flat plate segmentation tool in space claim for segmentation.



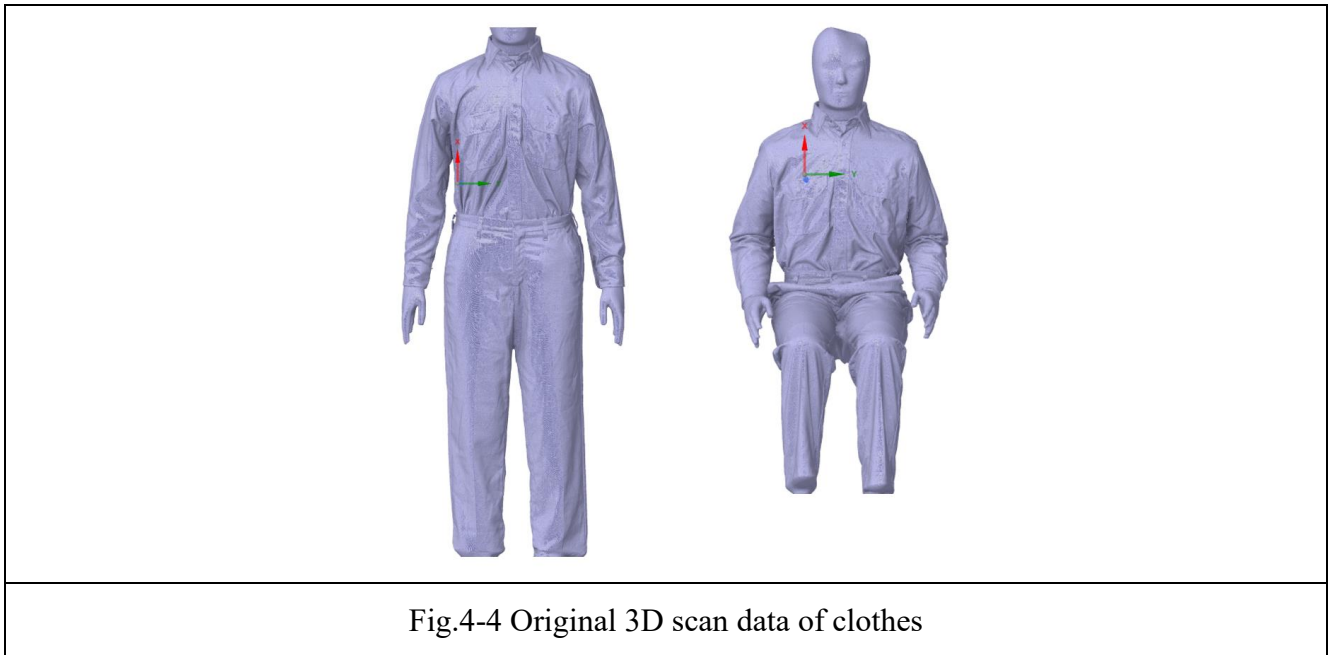
4.2.2 Dressing Models

Indoor clothing has a great influence on thermal comfort, especially for the non-steady indoor environment simulation including the human thermal model, the simulation of indoor clothing is of great significance. But fashion not only has a warm effect but also has a strong decorative attribute. Different people wear clothes, due to the influence of personal preferences and actual needs, the number of layers of clothing, clothing materials, and clothing tailoring will be very different. However, the essential function of clothing is to isolate the internal and external flow fields. Since air is a good human insulator, it can improve thermal resistance and reduce the heat loss of the human body.

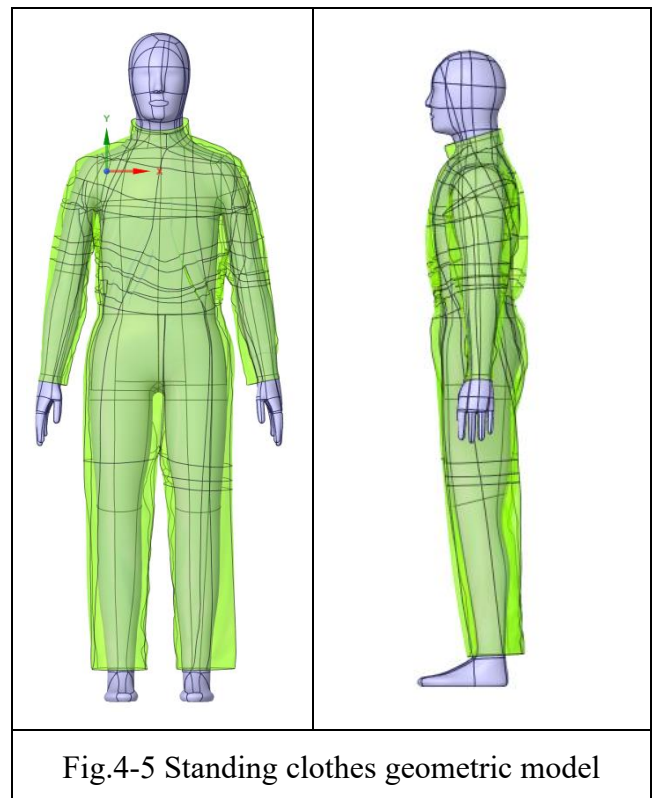
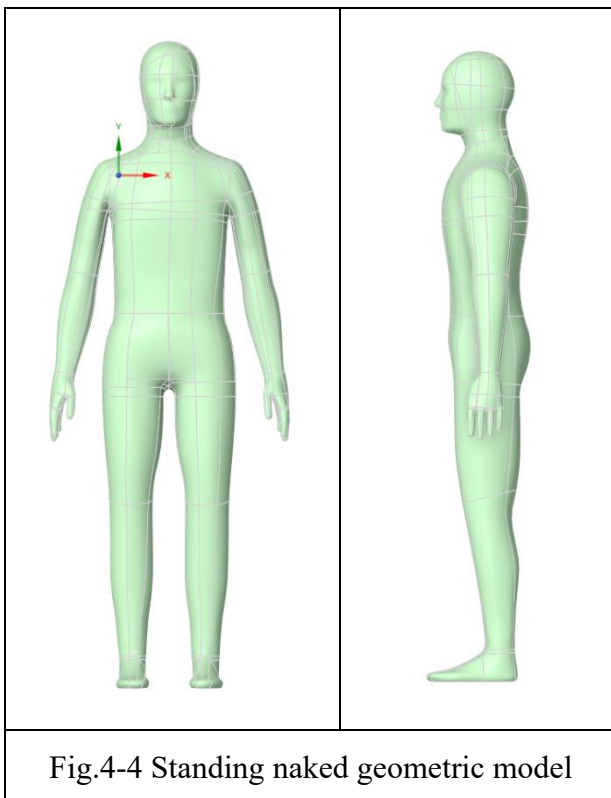
The size data of clothes used in this study are shown in the figure below,

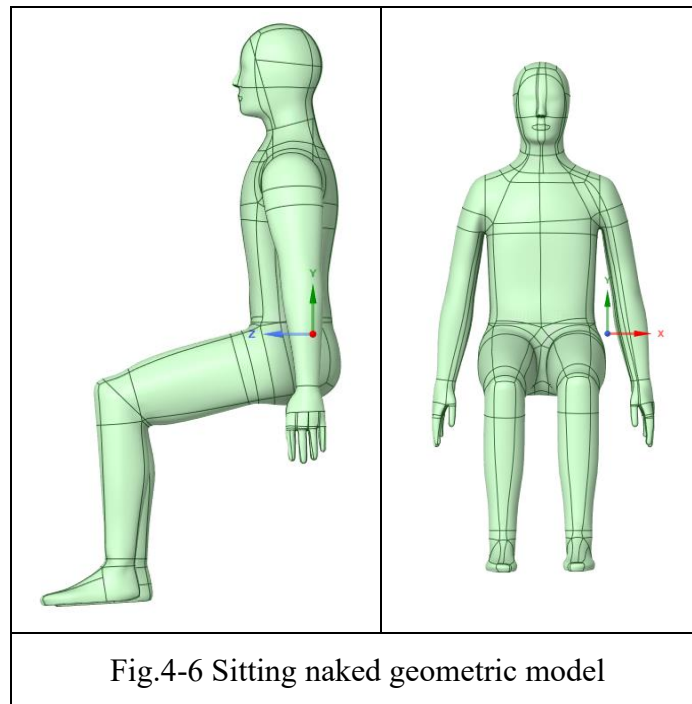
			
Shoulder width	45	Waist	41cm
Width	55	Hip	53cm
Sleeve length	59	Hem-width	22.5cm
Sleeve	24	The rise	22.5cm
Length	80	Inseam	81.5cm
		Thickness of thigh	35cm
Maker: BULWARK SIZE: M		Maker: BULWARK SIZE:33	
Fig.4-5 Measurement clothes			

Due to the complexity of the folds on the clothes, a one-time overall scan is required when performing a 3D scan. The obtained CAD clothing geometric model can reflect the overall outline of the clothing. However, since the complex structure inside the garment cannot be obtained by external 3D laser scanning, this study carried out geometric simplification on the collar, pockets, cuffs and other parts of the garment. Since the geometric data of the overscan needs to be obtained by collecting the reflected lattice laser information, the geometric information of the underarms, between the legs, and the parts that are in close contact with the cloth pieces cannot be directly obtained. Simplification is also required. Fig.4-4 shows the original 3D scan data of clothes.



The processed digital geometric model is shown in Fig.4-4, Fig.4-5 and Fig.4-6.





5 Validation of Digital Twin Coupled with Computational Fluid Dynamics

5.1 introduction

To verify the performance of Digital Twin in this study, the experimental environment built will be briefly introduced in 5.1.1. In the simulation, the same digital model room as the experimental environment is established, and the corresponding boundary condition settings are selected. The specific settings will be introduced in 5.2.2.

5.1.1 In-chamber Measurement

Thermal manikin developed by Thermetric has excellent performance (detailed introduction). This study placed it in a chamber to control the indoor thermal environment by adjusting the air conditioning system. Controlled main Supply air temperature.

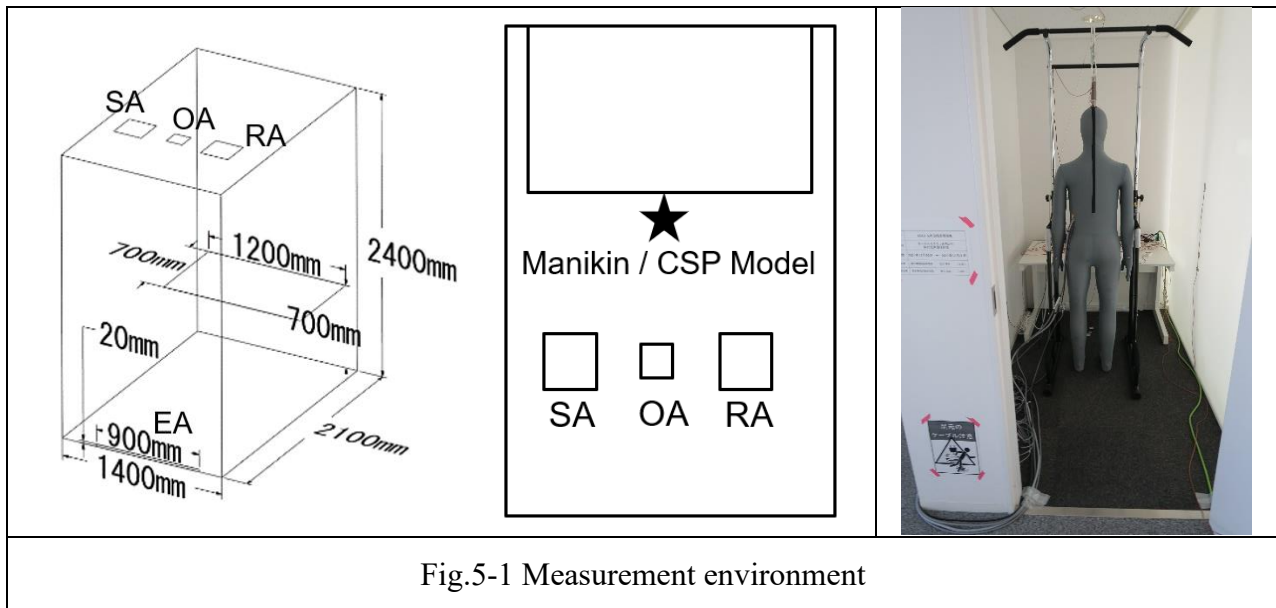


Fig.5-1 Measurement environment

The size of the Chamber is shown in the left figure of Fig.5-1. SA is supply air, RA is return air, and the set air volume of SA and RA is $164\text{m}^3/\text{h}$. OA is outer air with a flow rate of $47\text{m}^3/\text{h}$. The ambient temperature in the chamber is jointly controlled by OA and SA, and the temperature of RA is regarded as the control point of indoor temperature. EA is exhaust air, which was originally the gap between the door and the floor in the chamber to maintain indoor pressure. The picture in the middle of Fig.5-1 shows the placement of thermal manikin. The actual experimental environment is shown in the right picture of Fig.5-1.

In the experiment, the hourly temperatures of SA, OA and RA will be recorded, and the temperature of the human body surface will also be recorded. At the same time, the hourly temperature changes of the walls are also recorded. After the 30-min indoor measurement, the body surface temperature distribution of the thermal manikin will be recorded by the thermal imaging camera.

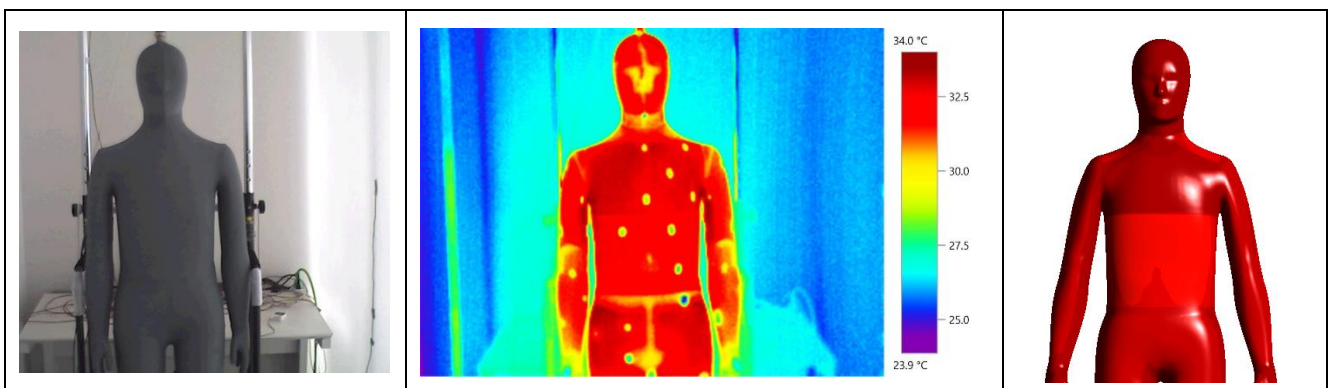
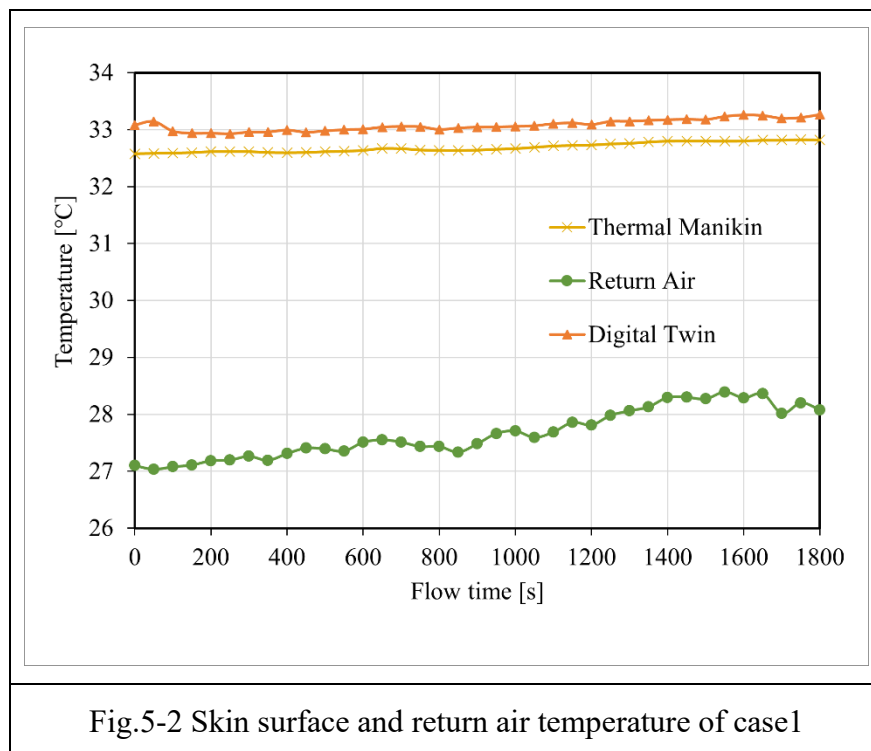
5.1.2 CFD software simulation

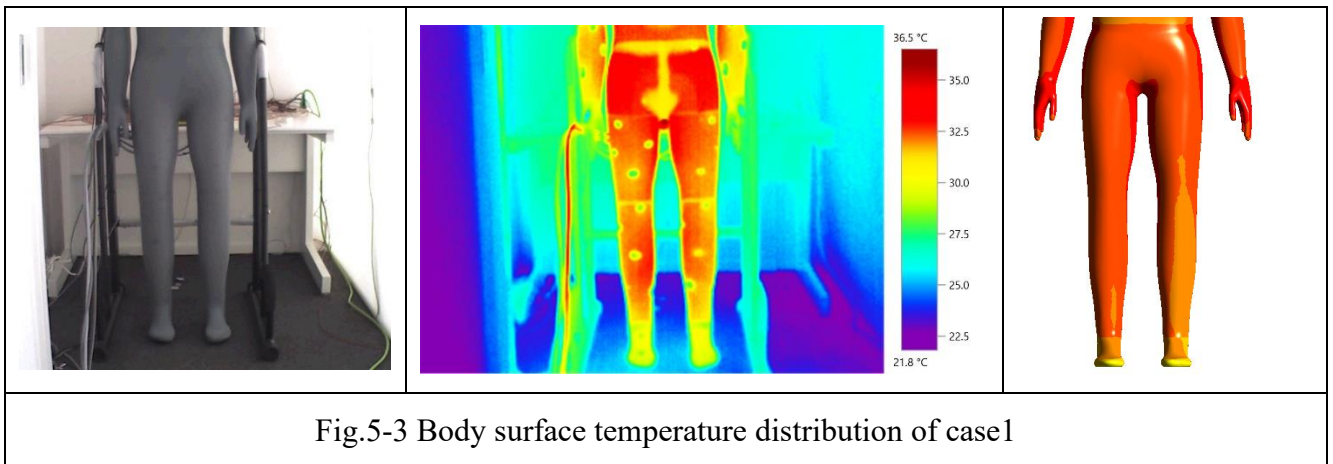
The room size in the simulation calculation and the position of the CSP model can also refer to Fig.5-1. As the boundary conditions, the air velocity and temperature settings of SA, OA, RA select the measured hourly data. The temperature of the wall also adopts the hourly data obtained by the actual measurement.

5.2 Simulation result of Digital twin of Thermal manikin

The human body thermal model that can be used for CFD calculation developed based on the UDF function of ANSYS FLUENT in Chapter 3 and the Thermal manikin obtained by 3D scanning in Chapter 4 are integrated. Computer computer-based CFD simulations can obtain a virtual human thermal model (digital twin) with approximate performance.

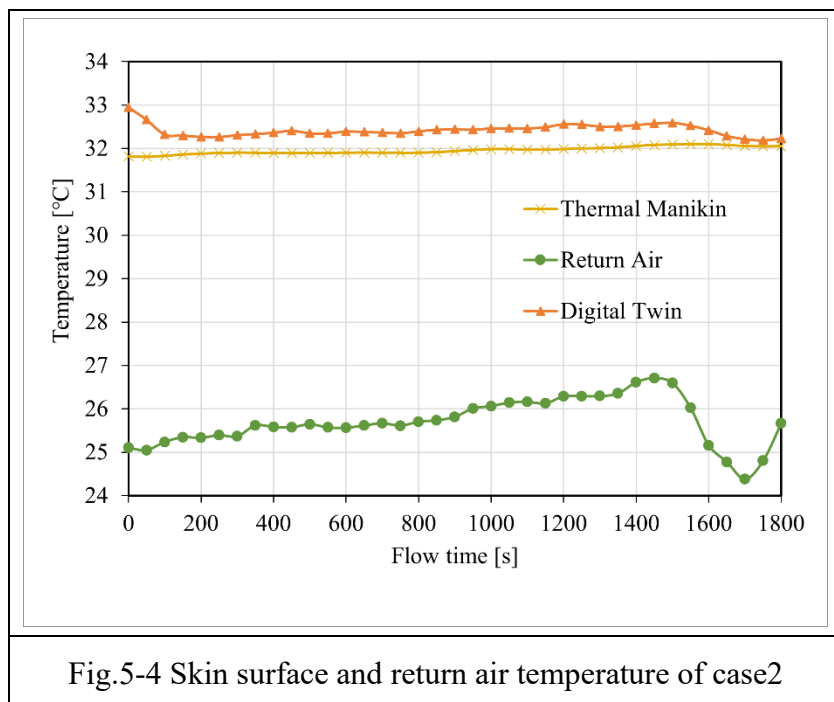
Case1: standing without clothes (28°C)

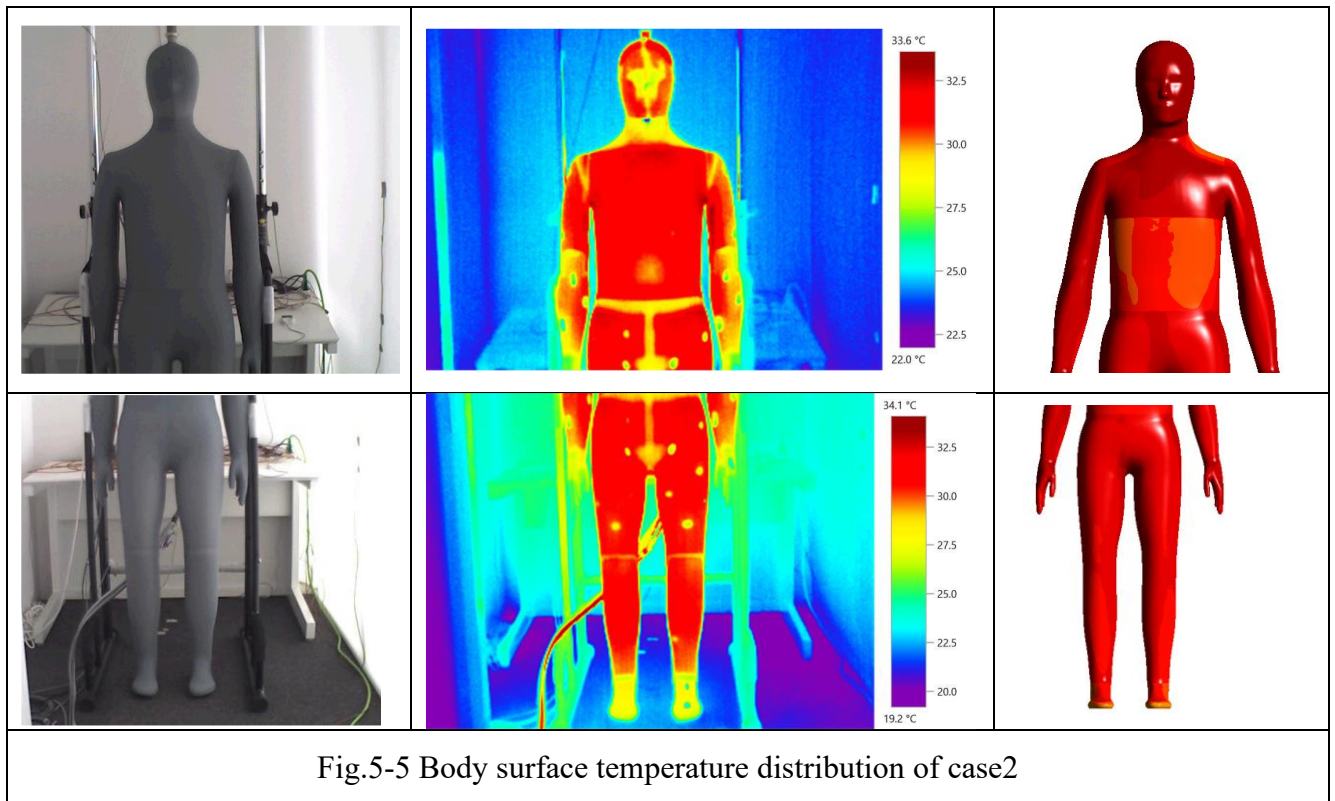




When the ambient temperature is 28 °C, the hourly temperature changes of the Digital Twin and Thermal Manikin are shown in Fig.5-2, and the two have the same trend of change with the ambient temperature within 30 minutes. The average temperature difference between the body surfaces is 0.3 °C. Fig.5-3 reflects the body surface temperature distribution of Digital twin and Thermal Manikin.

Case2: standing without clothes (24°C)





When the ambient temperature is 24 °C, the hourly temperature changes of the Digital Twin and Thermal Manikin are shown in Fig.5-4, and the two have the same trend of change with the ambient temperature within 30 minutes. The average temperature difference between the body surfaces is 0.3 °C. Fig.5-5 reflects the body surface temperature distribution of Digital twin and Thermal Manikin.

Case3: sitting without clothes (28°C)

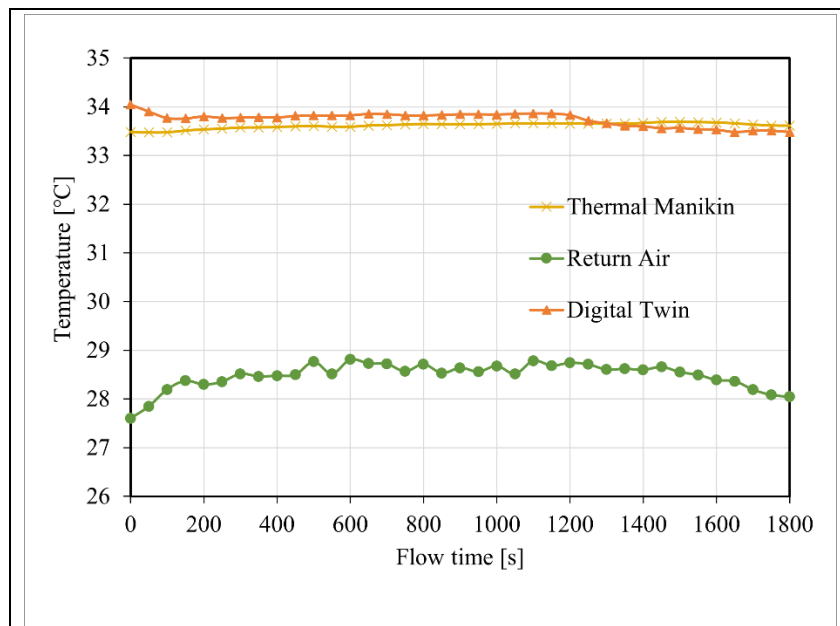
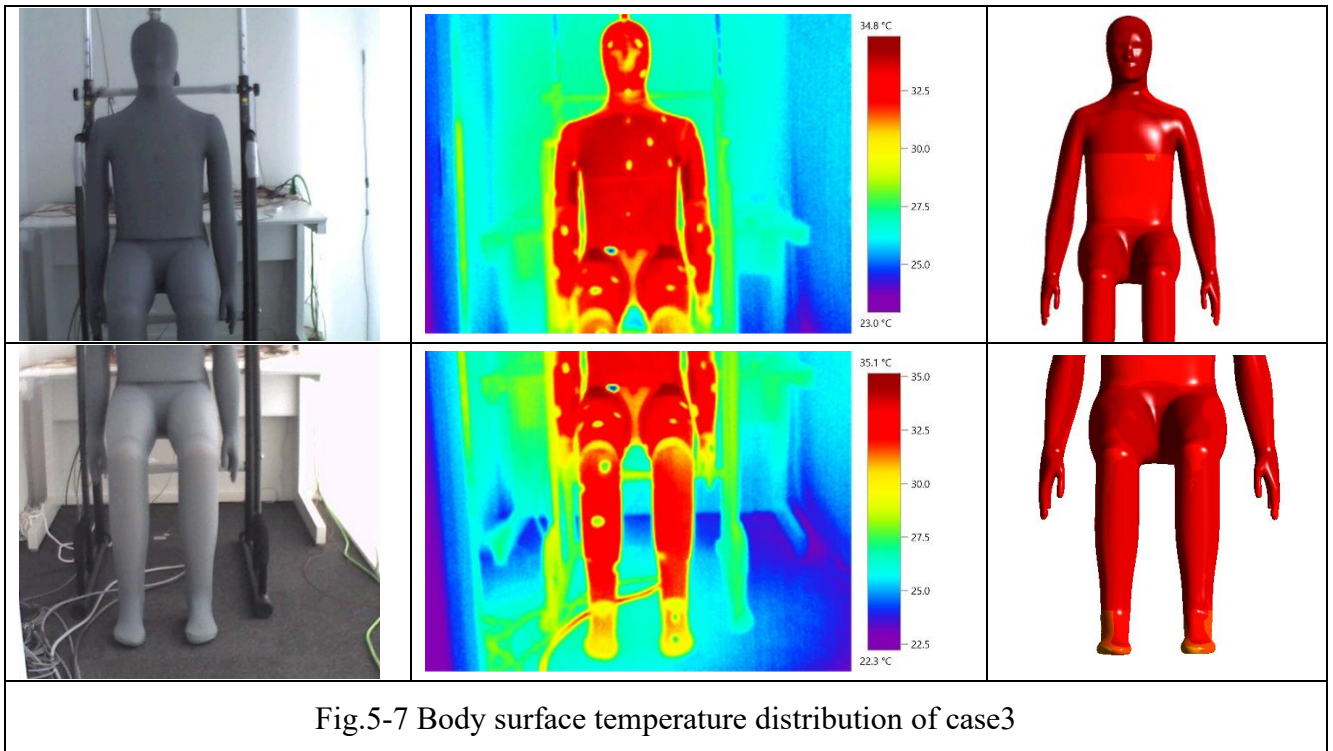
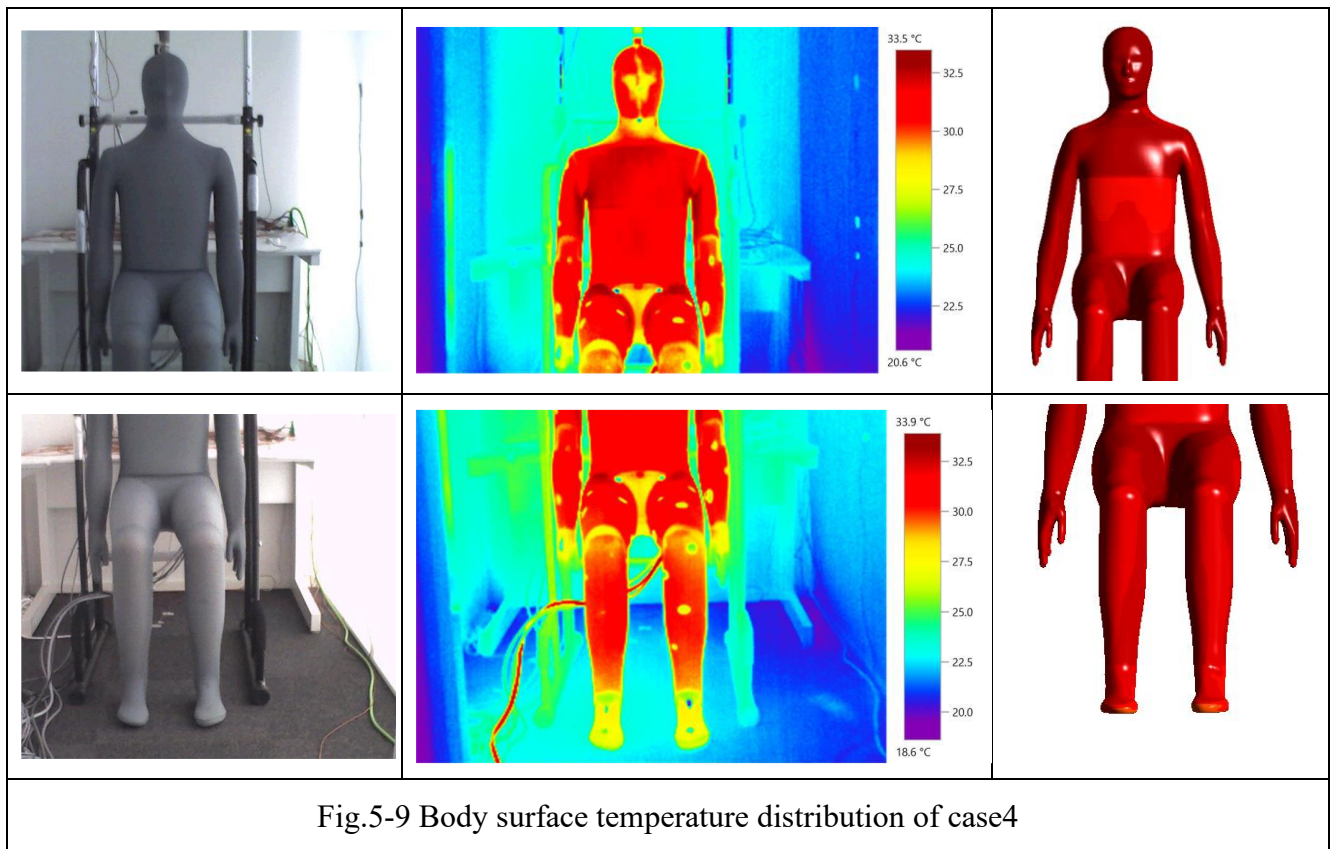
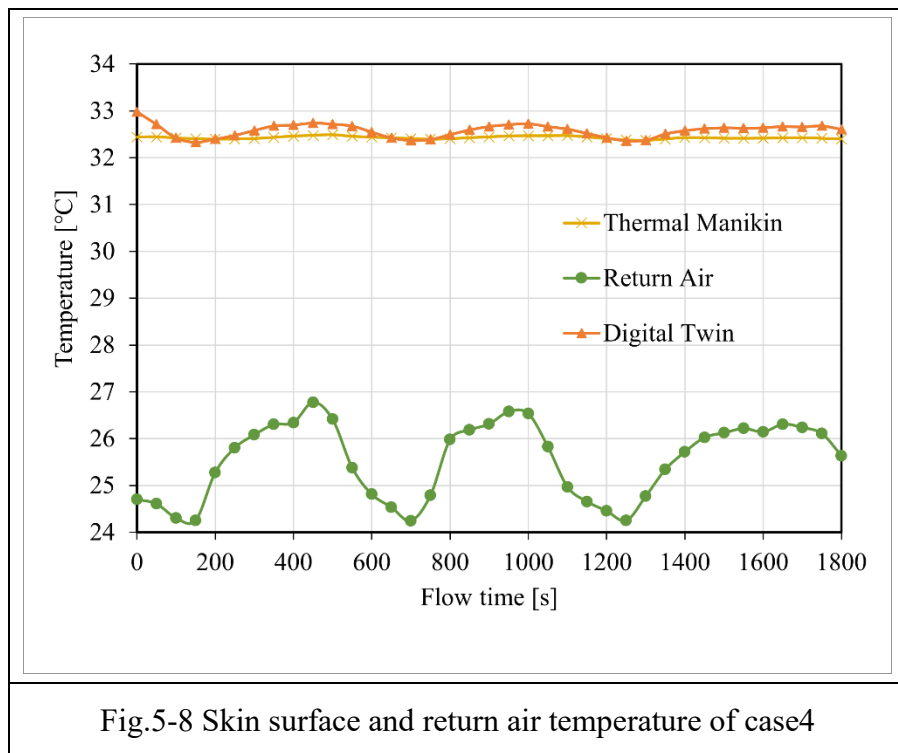


Fig.5-6 Skin surface and return air temperature of case3



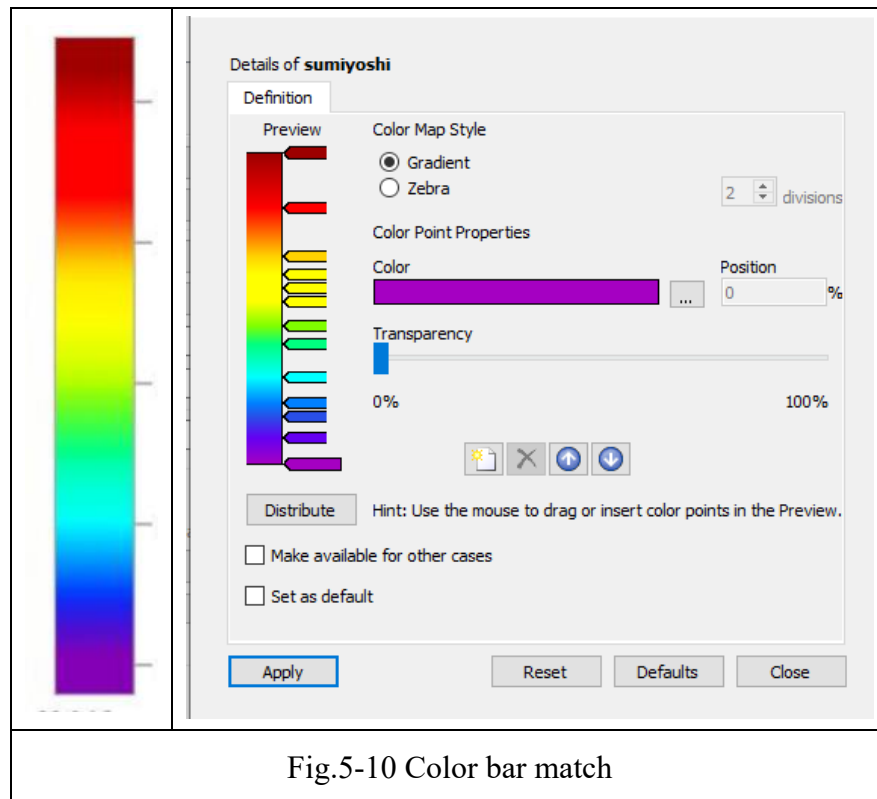
When the ambient temperature is 28 °C, the hourly temperature changes of the Digital Twin and Thermal Manikin are shown in Fig.5-6, and the two have the same trend of change with the ambient temperature within 30 minutes. The average temperature difference between the body surfaces is 0.2 °C. Fig.5-7 reflects the body surface temperature distribution of Digital twin and Thermal Manikin.

Case4: sitting without clothes (24°C)



When the ambient temperature is 24 °C, the hourly temperature changes of the Digital Twin and Thermal Manikin are shown in Fig.5-8, and the two have the same trend of change with the ambient temperature

within 30 minutes. The average temperature difference between the body surfaces is 0.15 °C. Fig.5-9 reflects the body surface temperature distribution of Digital twin and Thermal Manikin.



The color bar used for thermal infrared scanning of the Thermal Manikin body surface using a handheld device does not exist in ANSYS FLUENT POST, so manual matching is required. So that the color bar in the software is comparable with the measured body surface temperature distribution. Fig.5-10 is the operation interface when the color is matched.

6 Conclusion

In this study, a handheld 3D high-precision laser scanner was used to establish the same geometric model as the Thermal Manikin, and based on the USER DEFINE FUNCTION function of ANASY FLUENT, the same human body thermal model (Fiala Model) was established as the Thermal Manikin. After integrating the digital geometric model with the human body thermal model, the developed CSP model can be regarded as the Digital Twin of Thermal Manikin. Finally, under the same environmental

conditions, the Thermal Manikin was measured and the Digital was simulated by CFD, and the results were verified by comparing the two.

Through the final comparison, we found that the Digital Twin can well reflect the performance of Thermal Manikin. For example, the skin temperature increases or decreases in an unsteady environment, and different temperature distributions appear in different parts of the body. These characteristics that can more realistically reflect the thermal regulation of the human body help us to have a more detailed and real understanding of the heat dissipation of the human body when conducting indoor CFD research, and can also use the body surface temperature data that is closer to the real human body. A more precise assessment of comfort.

REFERENCES

- [1] A. Dynamic and T. Manikin, “Operator ’ s Manual,” 2019.
- [2] D. Fiala, “Dynamic Simulation of Human Heat Transfer and Thermal Comfort,” *Sustain. Dev.*, vol. 45, no. 2001, p. 1, 1998, [Online].
- [3] Ansys Inc, “Ansys Fluent v17 - Customization Manual,” vol. 15317, no. January, pp. 724–746, 2016.
- [4] NHK放送文化研究所, *国民生活時間調査 2005*. 日本放送出版協会, 2006.
- [5] L. G. Gagge, A. P., Fobelets, A. P. and Berglund, “A standard predictive Index of human reponse to thermal enviroment,” *Am. Soc. Heating, Refrig. Air-Conditioning Eng.*, vol. 92, no. 92(2B), pp. 709–731, 1986, [Online].
- [6] J. A. . Stolwijk, “A Mathematical Model of Physiological Temperature Regulation in Man,” *NASA Contract. Reports*, vol. 303, no. 1, pp. 14–30, 1971.
- [7] T. D. Canonsburg, “ANSYS FLUENT User ’ s Guide,” *Knowl. Creat. Diffus. Util.*, vol. 15317, no. October, pp. 724–746, 2012.
- [8] H. H. Pennes, “Analysis of tissue and arterial blood temperatures in the resting human forearm,”

J. Appl. Physiol., vol. 1, no. 2, pp. 93–121, 1948, doi: 10.5694/j.1326-5377.1946.tb34690.x.

- [9] M. J. E. vo. Wyndham, C. H., J. F. Morrison, C. G. Williams, N. B. Strydom and and A. H. Rahden, L. D. Holdsworth, A. J. Van Rensburg, A. Joffe, “Inter-and intra-individual differences in energy expenditure and mechanical efficiency,” *Ergonomics*, vol. 9, pp. 17–29, 1966.
- [10] E. C. Eberhart, *Heat Transfer in Medicine and Biology - Analysis and Applications*, vol. 1. New York and London: Plenum Press, 1985.
- [11] R. G. Gordon, “The Response of Human Thermoregulatory System in the Cold,” University of California, 1974.
- [12] and C. R. W. Rowell, L.B., *Temperature regulation in exercising and heat-stressed man*, vol. 1. New York and London: Plenum Press, 1985.

APPENDICES

A.1. The UDF macro code

```
1. #include "udf.h"
2. #include "math.h"
3. #include "mem.h"
4.
5. #define PI 3.14159
6.
7. /* XXX presents the name of the part xxx means the value */
8. /* Index ID of each part of human body */
9.
10. #define SKIN_XXX_ID xxx
11.
12. /* workload distribution coefficient */
13.
14. #define XXX_AM xxx
15.
16. /* Evaporation rate of each part [L/h] */
17.
18. #define XXX_VAPOR xxx
19.
```

```

20. /* hx CCX coefficient [W/K] */
21.
22. #define XXX_HX xxx
23.
24. /*length unit [m]*/
25.
26. #define XXX_L xxx
27.
28. /* outer radius, unit [m] */
29.
30. #define XXX_CORE_R xxx
31. #define XXX_MUSCLE_R xxx
32. #define XXX_FAT_R xxx
33. #define XXX_SKIN_R xxx
34.
35. /* heat conductivity, unit [W/(m*K)] */
36.
37. #define XXX_CORE_K xxx
38. #define XXX_MUSCLE_K xxx
39. #define XXX_FAT_K xxx
40. #define XXX_SKIN_K xxx
41.
42. /* material density, unit [kg/m^3] */
43.
44. #define XXX_CORE_RHO xxx
45. #define XXX_MUSCLE_RHO xxx
46. #define XXX_FAT_RHO xxx
47. #define XXX_SKIN_RHO xxx
48.
49. /* heat capacitance, unit [J/(kg*K)] */
50.
51. #define XXX_CORE_C xxx
52. #define XXX_MUSCLE_C xxx
53. #define XXX_FAT_C xxx
54. #define XXX_SKIN_C xxx
55.
56. /* blood perfusion, unit [ltr/(s*m^3)] */
57.
58. #define XXX_CORE_W xxx
59. #define XXX_MUSCLE_W xxx

```

```

60. #define XXX_FAT_W xxx
61. #define XXX_SKIN_W xxx
62.
63. /* metabolic rate in thermal neutrality, unit [W/m²] */
64.
65. #define XXX_CORE_Q0 xxx
66. #define XXX_MUSCLE_Q0 xxx
67. #define XXX_FAT_Q0 xxx
68. #define XXX_SKIN_Q0 xxx
69.
70. /*temperature set point of different compartments [K] */
71.
72. #define XXX_CORE_T0 xxx
73. #define XXX_MUSCLE_T0 xxx
74. #define XXX_FAT_T0 xxx
75. #define XXX_SKIN_T0 xxx
76.
77. /*number of nodes of each layer each node */
78.
79. #define XXX_CORE_NODE xxx
80. #define XXX_MUSCLE_NODE xxx
81. #define XXX_FAT_NODE xxx
82. #define XXX_SKIN_NODE xxx
83.
84. #define ACT 1
85.
86. #define VAPOR_PRESSURE_0 1735/*UDMI 45*/
87.
88. /*time step of calculation*/
89. #define dt 0.2
90.
91. DEFINE_INIT(XXX_RT, d)
92. {
93.     Thread* tcskin, * tskin;
94.     cell_t cskin;
95.     face_t fskin;
96.
97.     int Zone_ID = SKIN_XXX_ID;
98.     int nsum = XXX_CORE_NODE + XXX_MUSCLE_NODE + XXX_FAT_NODE + XXX_SKIN_NODE;
99.     int n;

```

```

100.    real rb[5], t0[5], no[5], r, t, dr;
101.
102.    /*number of nodes in each layer*/
103.    no[0] = XXX_CORE_NODE;
104.    no[1] = XXX_MUSCLE_NODE;
105.    no[2] = XXX_FAT_NODE;
106.    no[3] = XXX_SKIN_NODE;
107.
108.    /*radius of the boundary of each layer*/
109.    rb[0] = XXX_CORE_R;
110.    rb[1] = XXX_MUSCLE_R;
111.    rb[2] = XXX_FAT_R;
112.    rb[3] = XXX_SKIN_R;
113.
114.    /*temperature set point of each layer*/
115.    t0[0] = XXX_CORE_T0;
116.    t0[1] = XXX_MUSCLE_T0;
117.    t0[2] = XXX_FAT_T0;
118.    t0[3] = XXX_SKIN_T0;
119.
120.    /*find the cells adjacent to boundary XXX 7.0*/
121.    tskin = Lookup_Thread(d, Zone_ID);
122.
123.    /*initialization of temperature and radius of each node*/
124.    begin_f_loop(fskin, tskin)
125.    {
126.
127.        cskin = F_C0(fskin, tskin);
128.        tcskin = THREAD_T0(tskin);
129.
130.        Message("XXX cell\n");
131.
132.        for (n = 0; n <= nsum; n++)
133.        {
134.            if (n == 0)
135.            {
136.
137.                t = t0[0];
138.                r = 0.1 * rb[0];
139.

```

```

140.         }
141.         else if (n <= no[0])
142.         {
143.
144.             if (no[0] == 1)
145.             {
146.                 r = 0.7 * rb[0];
147.             }
148.             else
149.             {
150.                 dr = ((0.9 * rb[0]) * 2) / (no[0] * 2 + 1);
151.                 r = 0.1 * rb[0] + n * dr;
152.             }
153.             t = t0[0];
154.
155.         }
156.         else if (n <= no[0] + no[1])
157.         {
158.             if (no[1] == 1)
159.             {
160.                 r = 0.5 * (rb[1] + rb[0]);
161.             }
162.             else
163.             {
164.                 dr = (rb[1] - rb[0]) / no[1];
165.                 r = rb[0] + (n - no[0] - 0.5) * dr;
166.
167.             }
168.             t = t0[1];
169.         }
170.         else if (n <= no[0] + no[1] + no[2])
171.         {
172.             if (no[2] == 1)
173.             {
174.                 r = 0.5 * (rb[2] + rb[1]);
175.             }
176.             else
177.             {
178.                 dr = (rb[2] - rb[1]) / no[2];
179.                 r = rb[1] + (n - no[0] - no[1] - 0.5) * dr;

```

```

180.         }
181.         t = t0[2];
182.     }
183.     else if (n <= nsum)
184.     {
185.         if (no[3] == 1)
186.         {
187.             r = 0.5 * (rb[3] + rb[2]);
188.         }
189.         else
190.         {
191.             dr = (rb[3] - rb[2]) / no[3];
192.             r = rb[2] + (n - no[0] - no[1] - no[2] - 0.5) * dr;
193.         }
194.         t = t0[3];
195.     }
196.
197.     Message("n=%d,t=%g,r=%g\n", n, t, r);
198.
199.     C_UDMI(cskin, tcskin, n) = r;
200.     C_UDMI(cskin, tcskin, n + 20) = t;
201.     C_T(cskin, tcskin) = XXX_SKIN_T0;
202.     C_UDMI(cskin, tcskin, 41) = XXX_SKIN_T0;
203.     C_UDMI(cskin, tcskin, 45) = VAPOR_PRESSURE_0;
204.     C_UDMI(cskin, tcskin, 54) = 0;
205. }
206. }
207. end_f_loop(cskin, tskin)
208. }
209.
210. DEFINE_PROFILE(XXX_SKIN_TEMPERATURE, tskin, index)
211. {
212.
213.     Thread* tcskin;
214.     cell_t cskin;
215.     face_t fskin;
216.
217.     int n;
218.     int i;
219.     real sigbtv = 0, sigbtbv = 0, vr, bt;

```

```

220.    real tblp;
221.    real rskin = XXX_SKIN_R;
222.    real tffc;
223.    real skintemp;
224.    real vapor = XXX_VAPOR;
225.
226.    /*Total number of nodes*/
227.    real nsum = XXX_CORE_NODE + XXX_MUSCLE_NODE + XXX_FAT_NODE + XXX_SKIN_NODE;
228.
229.    real b_r[4], k_r[4], rho_r[4], c_r[4], d_r[4];
230.
231.    /*countercurrent heat exchange coefficient*/
232.    real hx = XXX_HX;
233.
234.    /* radius of each layer boundary*/
235.    b_r[0] = XXX_CORE_R;
236.    b_r[1] = XXX_MUSCLE_R;
237.    b_r[2] = XXX_FAT_R;
238.    b_r[3] = XXX_SKIN_R;
239.
240.    /*heat transfer coefficient of each layer*/
241.    k_r[0] = XXX_CORE_K;
242.    k_r[1] = XXX_MUSCLE_K;
243.    k_r[2] = XXX_FAT_K;
244.    k_r[3] = XXX_SKIN_K;
245.
246.    /*density of each layer*/
247.    rho_r[0] = XXX_CORE_RHO;
248.    rho_r[1] = XXX_MUSCLE_RHO;
249.    rho_r[2] = XXX_FAT_RHO;
250.    rho_r[3] = XXX_SKIN_RHO;
251.
252.    /*heat capacity of each layer*/
253.    c_r[0] = XXX_CORE_C;
254.    c_r[1] = XXX_MUSCLE_C;
255.    c_r[2] = XXX_FAT_C;
256.    c_r[3] = XXX_SKIN_C;
257.
258.    /*distance to adjacent nodes*/
259.    d_r[0] = (0.9 * b_r[0] * 2) / (XXX_CORE_NODE * 2 + 1);

```

```

260.    d_r[1] = (b_r[1] - b_r[0]) / XXX_MUSCLE_NODE;
261.    d_r[2] = (b_r[2] - b_r[1]) / XXX_FAT_NODE;
262.    d_r[3] = (b_r[3] - b_r[2]) / XXX_SKIN_NODE;
263.
264.    real r, k, rho, c, dr, tr0, tr1, tr, tskin;
265.
266.    /*parameters of front node and common parameter*/
267.    real rafc, rn, drn, drn1, dn, dsu, dn1, an, an1, afn, afn1, tn11, trn, trn1, kn, kn1, r
    n1;
268.    /*parameters of back node*/
269.    real tn00, trn0, abn0, abn, dn0, an0, kn0, rn0, drn0, dsu0;
270.    /*parameters for met calculation*/
271.    real dqm, dqmb, qmsh, qmb0, tset, qm, qb, qafc, qc, qsum;
272.    /*parameters for vapor calculation*/
273.    real pos, psk, eva;
274.    /*parameters for work load calculation*/
275.    real eta, H, qmw = 0, am = XXX_AM;
276.
277.    real tbla, btr, w;
278.
279.    real tnew[20];
280.
281.    real heatflux_o, heat;
282.    real area;
283.
284.    real farea[ND_ND];
285.    real heatflux_i;
286.
287.    real evp;
288.
289.    begin_f_loop(fskin, tskin)
290.    {
291.        cskin = F_C0(fskin, tskin);
292.        tcskin = THREAD_T0(tskin);
293.
294.        /*calculation of sigma betar*vr */
295.        for (n = 1; n <= nsum; n = n + 1)
296.        {
297.
298.            /*input the t and r from mem*/

```

```

299.         r = C_UDMI(cskin, tcskin, n);
300.         tr = C_UDMI(cskin, tcskin, n + 20);
301.         if (r < b_r[0])
302.         {
303.             if (n == 1)
304.             {
305.                 drn = 1.5 * d_r[0];
306.             }
307.             else
308.             {
309.                 drn = d_r[0];
310.             }
311.             w = XXX_CORE_W;
312.             tset = XXX_CORE_T0;
313.             qmb0 = XXX_CORE_Q0;
314.         }
315.         else if (r > b_r[0] && r < b_r[1])
316.         {
317.             drn = d_r[1];
318.             w = XXX_MUSCLE_W;
319.             tset = XXX_MUSCLE_T0;
320.             qmb0 = XXX_MUSCLE_Q0;
321.
322.         }
323.         else if (r > b_r[1] && r < b_r[2])
324.         {
325.             drn = d_r[2];
326.             w = XXX_FAT_W;
327.             tset = XXX_FAT_T0;
328.             qmb0 = XXX_FAT_Q0;
329.         }
330.         else if (r > b_r[2] && r < b_r[3])
331.         {
332.             drn = d_r[3];
333.             w = XXX_SKIN_W;
334.             tset = XXX_SKIN_T0;
335.             qmb0 = XXX_SKIN_Q0;
336.         }
337.
338.         vr = 2 * PI * r * drn * XXX_L;

```

```

339.
340.         dqm = 0;
341.         if (r > XXX_CORE_R && r < XXX_MUSCLE_R)
342.         {
343.             dqmb = qmb0 * (pow(2, ((tr - tset) / 10)));
344.
345.             eta = 0.2 * tanh(0.39 * ACT - 0.6);
346.             H = 87 * ((ACT / 0.8) * (1 - eta) - 1);
347.             qmw = am * H / (XXX_L * PI * (XXX_MUSCLE_R * XXX_MUSCLE_R - XXX_CORE_R * XX
X_CORE_R));
348.
349.             dqm = dqmb + qmw;
350.         }
351.         qm = dqm + qmb0;
352.         btr = dqm * 0.932 + 1069 * 3650 * w;
353.
354.         sigbtv = sigbtv + btr * vr;
355.         sigtbtv = sigtbtv + btr * vr * tr;
356.     }
357.     /*calculation of temperature of each layer */
358.     for (n = 1; n <= nsum; n = n + 1)
359.     {
360.         /*input the t and r from memory*/
361.         r = C_UDMI(cskin, tcskin, n);
362.         tr = C_UDMI(cskin, tcskin, n + 20);
363.
364.         /*calculation of necessary parameters for each node*/
365.         if (r < b_r[0])
366.         {
367.             rho = rho_r[0];
368.             c = c_r[0];
369.             k = k_r[0];
370.             tset = XXX_CORE_T0;
371.             qmb0 = XXX_CORE_Q0;
372.             w = XXX_CORE_W;
373.             dr = d_r[0];
374.
375.             /*calculation of temperature of last node or imaginary node*/
376.             if (r - 0.1 * b_r[0] > dr)
377.             {

```

```

378.         tr0 = C_UDMI(cskin, tcskin, n + 20 - 1);
379.     }
380.     else if (r - 0.1 * b_r[0] <= dr)
381.     {
382.         tr0 = tset;
383.     }
384.
385.     /*calculation of temperature of next node or imaginary node*/
386.     if (b_r[0] - r > dr)
387.     {
388.         tr1 = C_UDMI(cskin, tcskin, n + 20 + 1);
389.     }
390.     else if (b_r[0] - r < dr)
391.     {
392.         rafc = b_r[0];
393.         rn = r;
394.         rn1 = C_UDMI(cskin, tcskin, n + 1);
395.         drn = d_r[0];
396.         drn1 = d_r[1];
397.         trn = C_UDMI(cskin, tcskin, n + 20);
398.         trn1 = C_UDMI(cskin, tcskin, n + 20 + 1);
399.         kn = k_r[0];
400.         kn1 = k_r[1];
401.         /******
402.         dn = (log(rafc / rn)) / (log((rn + drn) / rn));
403.         dsn = (log(rafc / (rn + drn))) / (log((rn + drn) / rn));
404.         dn1 = (log(rafc / rn1)) / (log(rn1 / (rn1 - drn1)));
405.
406.         an = kn / (log((rn + drn) / rn));
407.         an1 = kn1 / (log(rn1 / (rn1 - drn1)));
408.
409.         afn = (dsn * an1 - dn1 * an) / (dn * an1 - dn1 * an);
410.         afn1 = an1 / (dn * an1 - dn1 * an);
411.
412.         tn11 = trn * afn + trn1 * afn1;
413.         /******
414.         tr1 = tn11;
415.     }
416. }
417. else if (r > b_r[0] && r < b_r[1])

```

```

418.         {
419.             rho = rho_r[1];
420.             c = c_r[1];
421.             k = k_r[1];
422.             tset = XXX_MUSCLE_T0;
423.             qmb0 = XXX_MUSCLE_Q0;
424.             dr = d_r[1];
425.             w = XXX_MUSCLE_W;
426.
427.             /*calculation of temperature of last node or imaginary node*/
428.             if (r - b_r[0] > dr)
429.             {
430.                 tr0 = C_UDMI(cskin, tcskin, n + 20 - 1);
431.             }
432.             else if (r - b_r[0] < dr)
433.             {
434.                 rafc = b_r[0];
435.                 rn = r;
436.                 rn0 = C_UDMI(cskin, tcskin, n - 1);
437.                 drn = d_r[1];
438.                 drn0 = d_r[0];
439.                 trn = C_UDMI(cskin, tcskin, n + 20);
440.                 trn0 = C_UDMI(cskin, tcskin, n + 20 - 1);
441.                 kn = k_r[1];
442.                 kn0 = k_r[0];
443.                 /******
444.                 dn = (log(rafc / rn)) / (log(rn / (rn - drn)));
445.                 dsn = (log(rafc / (rn - drn))) / (log(rn / (rn - drn)));
446.                 dn0 = (log(rafc / rn0)) / (log((rn0 + drn0) / rn0));
447.
448.                 an = kn / (log(rn / (rn - drn)));
449.                 an0 = kn0 / (log((rn0 + drn0) / rn0));
450.
451.                 abn = (dn0 * an - dsn * an0) / (dn0 * an - dn * an0);
452.                 abn0 = an0 / (dn0 * an - dn * an0);
453.
454.                 tn00 = trn0 * abn0 + trn * abn;
455.                 /******
456.                 tr0 = tn00;
457.             }

```

```

458.
459.      /*calculation of temperature of next node or imaginary node*/
460.      if (b_r[1] - r > dr)
461.      {
462.          tr1 = C_UDMI(cskin, tcskin, n + 20 + 1);
463.      }
464.      else if (b_r[1] - r < dr)
465.      {
466.          rafc = b_r[1];
467.          rn = r;
468.          rn1 = C_UDMI(cskin, tcskin, n + 1);
469.          drn = d_r[1];
470.          drn1 = d_r[2];
471.          trn = C_UDMI(cskin, tcskin, n + 20);
472.          trn1 = C_UDMI(cskin, tcskin, n + 20 + 1);
473.          kn = k_r[1];
474.          kn1 = k_r[2];
475.          /*****
476.          dn = (log(rafc / rn)) / (log((rn + drn) / rn));
477.          dsn = (log(rafc / (rn + drn))) / (log((rn + drn) / rn));
478.          dn1 = (log(rafc / rn1)) / (log(rn1 / (rn1 - drn1)));
479.
480.          an = kn / (log((rn + drn) / rn));
481.          an1 = kn1 / (log(rn1 / (rn1 - drn1)));
482.
483.          afn = (dsn * an1 - dn1 * an) / (dn * an1 - dn1 * an);
484.          afn1 = an1 / (dn * an1 - dn1 * an);
485.
486.          tn11 = trn * afn + trn1 * afn1;
487.          /*****
488.          tr1 = tn11;
489.      }
490.  }
491.
492.      else if (r > b_r[1] && r < b_r[2])
493.      {
494.          dr = d_r[2];
495.          rho = rho_r[2];
496.          c = c_r[2];
497.          k = k_r[2];

```

```

498.         w = XXX_FAT_W;
499.         tset = XXX_FAT_T0;
500.         qmb0 = XXX_FAT_Q0;
501.
502.         /*calculation of temperature of last node or imaginary node*/
503.         if (r - b_r[1] > dr)
504.         {
505.             tr0 = C_UDMI(cskin, tcskin, n + 20 - 1);
506.         }
507.
508.         else if (r - b_r[1] < dr)
509.         {
510.             rffc = b_r[1];
511.             rn = r;
512.             rn0 = C_UDMI(cskin, tcskin, n - 1);
513.             drn = d_r[2];
514.             drn0 = d_r[1];
515.             trn = C_UDMI(cskin, tcskin, n + 20);
516.             trn0 = C_UDMI(cskin, tcskin, n + 20 - 1);
517.             kn = k_r[2];
518.             kn0 = k_r[1];
519.             /******
520.             dn = (log(rffc / rn)) / (log(rn / (rn - drn)));
521.             dsn = (log(rffc / (rn - drn))) / (log(rn / (rn - drn)));
522.             dn0 = (log(rffc / rn0)) / (log((rn0 + drn0) / rn0));
523.
524.             an = kn / (log(rn / (rn - drn)));
525.             an0 = kn0 / (log((rn0 + drn0) / rn0));
526.
527.             abn = (dn0 * an - dsn * an0) / (dn0 * an - dn * an0);
528.             abn0 = an0 / (dn0 * an - dn * an0);
529.
530.             tn00 = trn0 * abn0 + trn * abn;
531.             /******
532.             tr0 = tn00;
533.         }
534.         /*calculation of temperature of next node or imaginary node*/
535.         if (b_r[2] - r > dr)
536.         {
537.             tr1 = C_UDMI(cskin, tcskin, n + 20 + 1);

```

```

538.
539.         }
540.         else if (b_r[2] - r < dr)
541.         {
542.             rafc = b_r[2];
543.             rn = r;
544.             rn1 = C_UDMI(cskin, tcskin, n + 1);
545.             drn = d_r[2];
546.             drn1 = d_r[3];
547.             trn = C_UDMI(cskin, tcskin, n + 20);
548.             trn1 = C_UDMI(cskin, tcskin, n + 20 + 1);
549.             kn = k_r[2];
550.             kn1 = k_r[3];
551.             /*****
552.             dn = (log(rafc / rn)) / (log((rn + drn) / rn));
553.             dsn = (log(rafc / (rn + drn))) / (log((rn + drn) / rn));
554.             dn1 = (log(rafc / rn1)) / (log(rn1 / (rn1 - drn1)));
555.
556.             an = kn / (log((rn + drn) / rn));
557.             an1 = kn1 / (log(rn1 / (rn1 - drn1)));
558.
559.             afn = (dsn * an1 - dn1 * an) / (dn * an1 - dn1 * an);
560.             afn1 = an1 / (dn * an1 - dn1 * an);
561.
562.             tn11 = trn * afn + trn1 * afn1;
563.             /*****
564.             tr1 = tn11;
565.         }
566.     }
567.
568.     else if (r > b_r[2] && r < b_r[3])
569.     {
570.         dr = d_r[3];
571.         rho = rho_r[3];
572.         c = c_r[3];
573.         k = k_r[3];
574.         w = XXX_SKIN_W;
575.         tset = XXX_SKIN_T0;
576.         qmb0 = XXX_SKIN_Q0;
577.

```

```

578.         /*calculation of temperature of last node or imaginary node*/
579.         if (r - b_r[2] < dr)
580.         {
581.             rafc = b_r[2];
582.             rn = r;
583.             rn0 = C_UDMI(cskin, tcskin, n - 1);
584.             drn = d_r[3];
585.             drn0 = d_r[2];
586.             trn = C_UDMI(cskin, tcskin, n + 20);
587.             trn0 = C_UDMI(cskin, tcskin, n + 20 - 1);
588.             kn = k_r[3];
589.             kn0 = k_r[2];
590.             /*****
591.             dn = (log(rafc / rn)) / (log(rn / (rn - drn)));
592.             dsn = (log(rafc / (rn - drn))) / (log(rn / (rn - drn)));
593.             dn0 = (log(rafc / rn0)) / (log((rn0 + drn0) / rn0));
594.
595.             an = kn / (log(rn / (rn - drn)));
596.             an0 = kn0 / (log((rn0 + drn0) / rn0));
597.
598.             abn = (dn0 * an - dsn * an0) / (dn0 * an - dn * an0);
599.             abn0 = an0 / (dn0 * an - dn * an0);
600.
601.             tn00 = trn0 * abn0 + trn * abn;
602.             /*****
603.             tr0 = tn00;
604.
605.             tr1 = C_UDMI(cskin, tcskin, n + 20 + 1);
606.         }
607.         else if (r - b_r[2] > dr)
608.         {
609.             if (b_r[3] - r > dr)
610.             {
611.
612.                 tr1 = C_UDMI(cskin, tcskin, n + 20 + 1);
613.                 tr0 = C_UDMI(cskin, tcskin, n + 20 - 1);
614.             }
615.
616.             else if (b_r[3] - r < dr)
617.             {

```

```

618.
619.             rn = r;
620.             rn0 = C_UDMI(cskin, tcskin, n - 1);
621.             rffc = 0.5 * (rn + rn0);
622.
623.             tr0 = C_UDMI(cskin, tcskin, n + 20 - 1);
624.             tr = C_UDMI(cskin, tcskin, n + 20);
625.
626.             /*calculation the heatflux under skin surface*/
627.
628.             F_AREA(farea, fskin, tskin);
629.             area = NV_MAG(farea);
630.
631.             heatflux_i = (k * (tr0 - tr) * area) / (log(rn / rn0) * 0.5 * (rn +
        rn0));
632.
633.             C_UDMI(cskin, tcskin, 42) = heatflux_i / area;
634.             /*calculation the heatflux on skin surface*/
635.
636.             heat = BOUNDARY_HEAT_FLUX(fskin, tskin);
637.
638.             heatflux_o = heat + 0.675 * vapor * area; /*vapor is flow set point
        of Thermal manikin*/
639.
640.
641.             C_UDMI(cskin, tcskin, 55) = evp;
642.
643.             C_UDMI(cskin, tcskin, 56) = pos;
644.
645.             C_UDMI(cskin, tcskin, 57) = psk;
646.
647.             C_UDMI(cskin, tcskin, 43) = heatflux_o / area;
648.             }
649.         }
650.     }
651.
652.     /*the calculation of met*/
653.
654.     dqm = 0;
655.     /*calculation of extra met in muscle layer*/

```

```

656.         if (r > XXX_CORE_R && r < XXX_MUSCLE_R)
657.         {
658.             /*calculation of the change in the basal metabolism*/
659.             dqmb = qmb0 * (pow(2, ((trn - tset) / 10)));
660.
661.             /*work term*/
662.             eta = 0.2 * tanh(0.39 * ACT - 0.6);
663.             H = 87 * ((ACT / 0.8) * (1 - eta) - 1);
664.             qmw = am * H / (XXX_L * PI * (XXX_MUSCLE_R * XXX_MUSCLE_R - XXX_CORE_R * XX
X_CORE_R));
665.
666.             dqm = dqmb + qmw;
667.         }
668.
669.         qm = dqm + qmb0;
670.
671.         /*the calculation of blood circulation*/
672.
673.         tblp = 37 + 273.15;
674.         tbla = ((tblp * sigbtv) / (hx + sigbtv)) + ((hx * sigbtv / sigbtv) / (hx + sig
btv));
675.
676.         C_UDMI(cskin, tcskin, 44) = tbla;
677.
678.         /*the calculation of temperature of next time step*/
679.
680.         btr = dqm * 0.932 + 1069 * 3650 * w;
681.
682.         qb = btr * (tbla - tr);
683.
684.         /*remove the qb and qm of outer skin*/
685.         if (r > (b_r[3] + b_r[2]) / 2)
686.         {
687.             qb = 0;
688.             qm = 0;
689.         }
690.
691.         if (b_r[3] - r > d_r[3])
692.         {
693.

```

```

694.         qc = k * ((tr1 + tr0 - 2 * tr) / (dr * dr) + (tr1 - tr0) / (2 * r * dr));
695.         qsum = qc + qb + qm;
696.
697.         tnew[n] = (qsum * dt) / (rho * c) + tr;
698.     }
699.
700.     if (b_r[3] - r < d_r[3])
701.
702.     {
703.         qsum = (heatflux_i - heatflux_o) / (dr * area);
704.
705.         tnew[n] = (qsum * dt) / (rho * c) + tr;
706.
707.         skintemp = tnew[n] - ((heatflux_i / area) * r * log(rskin / r) / k);
708.
709.         C_UDMI(cskin, tcskin, 41) = skintemp;
710.
711.         F_PROFILE(fskin, tskin, index) = C_UDMI(cskin, tcskin, 41);
712.     }
713.
714.     /*the calculation of skin temperature*/
715. }
716. /*input of the temp*/
717. for (n = 1; n <= nsum; n = n + 1)
718. {
719.     C_UDMI(cskin, tcskin, n + 20) = tnew[n]; /*temperature saving*/
720.
721. }
722. }
723. end_f_loop(fskin, tskin)
724. }
725.
726. DEFINE_PROFILE(XXX_VVAPOR, tskin, i)
727. {
728.
729.     Thread* tcskin;
730.     cell_t cskin;
731.     face_t fskin;
732.
733.     real x[ND_ND];

```

```

734.    real y[ND_ND];
735.    real dists, dist, dx;
736.    real psk, dair, dsk, pos;
737.    real L;
738.
739.    real a;
740.    real diff;
741.
742.    begin_f_loop(fskin, tskin)
743.    {
744.        cskin = F_C0(fskin, tskin);
745.        tcskin = THREAD_T0(tskin);
746.        F_CENTROID(x, fskin, tskin);
747.        C_CENTROID(y, cskin, tcskin);
748.
749.        dist = pow(x[0] - y[0], 2) + pow(x[1] - y[1], 2) + pow(x[2] - y[2], 2);
750.        dists = sqrt(dist);
751.        dx = dists;
752.
753.        L = C_UDMI(cskin, tcskin, 46);
754.
755.        dair = C_UDSI(cskin, tcskin, 0);
756.
757.        diff = C_UDSI_DIFF(cskin, tcskin, 0);
758.
759.        dsk = (L * dx) / (diff * 2256000) + dair;
760.
761.        psk = dsk * 101325 / 0.622;
762.
763.        C_UDMI(cskin, tcskin, 45) = psk;
764.
765.        C_UDMI(cskin, tcskin, 51) = dair;
766.        C_UDMI(cskin, tcskin, 52) = dsk;
767.        C_UDMI(cskin, tcskin, 53) = dx;
768.        C_UDMI(cskin, tcskin, 54) = diff;
769.
770.        F_PROFILE(fskin, tskin, i) = dsk;
771.    }
772.    end_f_loop(fskin, tskin)
773. }

```

